United States Coast Guard

Department of Transportation

Year 2000 Management Plan

Version 2.0 November 1998

Commandant (G-SI/Y2K)
United States Coast Guard Headquarters

COMDTPUB P5230.9 **http://www.uscg.mil/systems/library/y2k.htm**

U.S. Department of Transportation

United States Coast Guard

Commandant
United States Coast Guard

2100 Second Street S.W.
Washington, DC 20593-0001
Staff Symbol: G-SI/Y2K
Phone: (202) 267-1275

COMDTPUB P5230.9

20 NOV 1998

COMMANDANT PUBLICATION P5230.9

Subj: YEAR 2000 MANAGEMENT PLAN

Ref: (a) Year 2000 Impact On Information Systems, COMDTINST 5230.3
(b) Year 2000 Compliance Reporting Requirements, COMDTINST 5230.7

1. PURPOSE. This publication provides Year 2000 guidance for "Y2K problem" correction of mission critical and non-mission critical systems. The "Y2K problem" is the term used to describe the potential failure of information technology prior to, on or after 1 January, 2000.

2. ACTION. Area and district commanders, commanders of maintenance and logistics commands, commanding officers of headquarters units, assistant commandants for directorates, Chief Counsel, and special staff offices at Headquarters shall ensure that their unit commanders are aware of the contents of this publication and are provided printed distribution via Acrobat Reader at the following internet address; **http://www.uscg.mil/systems/library/y2k.htm**

3. PUBLICATIONS AFFECTED. Coast Guard Year 2000 Management Plan of July 1997 is superseded.

4. DISCUSSION. Reference (a) provided initial awareness and policy for Year 2000 impact. COMDTPUB 5230.9 provides the overall strategy and management approach to satisfactorily address Year 2000 date processing problems within the Coast Guard. Areas of revision/expansion include extensive testing guidance, clarifying language and web site addresses which provide guidance in areas other than systems. Reference (b) consolidates all Year 2000 reporting requirements.

[illegible]

G.N. NACCARA
Director of Information and Technology

Encl: (1) Year 2000 Management Plan, Version 2.0, November 1998

DISTRIBUTION – SDL No.135

	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
B	1	8	10	1	12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
C	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
D	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
E	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
F																	1	1	1							
G	1	1	1	1	1																					
H																										

NON-STANDARD DISTRIBUTION:

Preface

The information in this document contains no sensitive material and should it be useful to your Year 2000 (Y2K) effort it may be freely copied. It may be viewed or printed via Acrobat Reader at the following Internet address "**http://www.uscg.mil/systems/library/y2k.htm**". Guidelines in this document are expressed in terms of potential problems and does not ensure a complete Y2K resolution.

The Y2K date processing problem is the most significant event to face world business organizations since the Great Depression. The Y2K issue poses an enormous management and technical challenge for the Coast Guard as well. The problem stems from the use of two-digit year fields instead of four-digit year fields in software, hardware, and embedded microchips. The effect of this will be that many computer programs and pieces of equipment will fail as they attempt to perform calculations and sorting routines because the systems and microchips will interpret the "00" as "1900" instead of "2000". The resulting inaccuracies in date-related calculations and sorting routines will generate corrupt data results and potentially cause systems to fail entirely. Also, if erroneous information goes unrecognized, the problem is perpetuated through interfaces with other automated information systems. While this is the crux of the problem, it is more complex. Many systems have faulty date logic that does not recognize that the year 2000 is a leap year, other systems have triggers that are executed based on specific values of date fields, and others have overflow or rollover problems. Finally, Y2K problems are happening today and could increase in frequency as we approach the year 2000.

The Y2K problem is not restricted to any one functional area within the Coast Guard. We use computers and microprocessor control systems to support every facet of our business. They perform or support the performance of our strategic and tactical operations such as mobilizing, deploying, and maneuvering resources. Computers are used to support intelligence, surveillance, and security efforts. Also, the Coast Guard relies on computers to support functions such as financial management, personnel management, health care, contract management, logistics management, and many other business functions. The Y2K problem stretches beyond our automated information systems (AIS) noted above. It can have serious effect on our infrastructure, most notably on control systems such as those that affect heating, ventilation, air conditioning and, of course, other embedded microprocessor systems. World Wide Web site addresses contained in this publication provide supplemental guidance in areas outside the Coast Guard functional areas of primary concern. In view of the magnitude and seriousness of the problem, the Coast Guard must assure that systems and equipment supporting operational missions and decision-making functions continue to perform as designed.

This Year 2000 (Y2K) Management Plan provides the strategic guidance for all information technology, software and systems in the United States Coast Guard that face a "Y2K problem." This document provides the overall strategy and management approach to satisfactorily address the Year 2000 date processing problem within Coast Guard systems. The Plan outlines responsibilities and provides guidelines for Year 2000 activities to ensure that no Coast Guard system fails due to Y2K problems. Version 1.0 of the Plan was released in limited distribution in June of 1997. This Version, 2.0, is released for Coast Guard-wide distribution and has been expanded to include additional guidance, testing guidelines, revised milestones and the guidelines for non-mission critical systems.

Table of Contents

I. Introduction

1.1 This Year 2000 (Y2K) Management Plan provides guidance for Coast Guard Headquarters Program Managers, and as appropriate, Headquarters units, Area Commanders, Commanders, Maintenance and Logistics Commands, and District Commanders, in the conduct of their Year 2000 efforts. The Management Plan also provides the strategic guidance for all information technology, software and systems in the United States Coast Guard (Coast Guard) that face a "Y2K problem." The "Y2K problem" is the term used to describe the potential failure of information technology (IT) prior to, on or after January 1, 2000. This potential exists because of the widespread practice of using two digits, not four, to represent the year in computer databases, software applications, and hardware chips. Y2K related difficulties will arise when that year is "00" and the information technology will be unable to differentiate the year 2000 from the year 1900. The associated but unrelated calendar anomaly that must also be included in Y2K system repairs is the fact that the year 2000 is an unusual century leap year.

1.2 The Director of Information and Technology (G-SI), as the Coast Guard's Chief Information Officer (CIO), has the responsibility to lead Coast Guard efforts in solving Y2K problems. Commandant (G-SI/Y2K), the Y2K Staff, will facilitate information flow, coordinate data call submissions, and connect units with Y2K solution providers as needed. Headquarters Program Managers and Coast Guard unit commanders are responsible for internal awareness, assessments, renovations, validations, and implementation actions. Chapter VII of this guide provides an overview of Coast Guard responsibilities.

1.3 The Coast Guard's goal is a service-wide effort to minimize the impact of the Y2K problem on the Coast Guard. To assist in this effort the CIO has adopted GAO's Year 2000 "Best Practices" five phase approach**: Awareness, Assessment, Renovation, Validation, and Implementation,** supported by program and project management activities. This approach will allow units the flexibility to implement solutions as deemed appropriate while benefiting from best practices in a coordinated effort While the CIO has ultimate oversight over the Coast Guard's Y2K repair program, **responsibility for renovating systems or pieces of equipment rests with owners all the way to the unit level. Unit commanders** are responsible for ensuring that all systems, software, and equipment at their commands correctly process dates, and should use the five phase approach or the equivalent to manage and document their Y2K efforts

1.4 Most experts agree that Y2K resolution efforts begin with a thorough assessment (inventory) of existing systems. Commandant (G-SI) has accumulated an extensive inventory of hardware and software which reflects more than three decades of IT development. Our goal is to have all Coast Guard systems certified as Y2K compliant and implemented not later than March 31, 1999. This will be accomplished through the elimination, replacement and/or modification of existing systems as they move through the phases. Supplemental guidance in areas of microprocessor control systems is contained on Web sites listed in Appendix F.

1.5 The new Federal Acquisition Regulations, Part 39.106 (FAR 39.106) states that compliance is: "information technology that accurately processes date/time data (including but not limited to, calculating, comparing, and sequencing) from, into, and between the twentieth and twenty-first centuries, and the years 1999 and 2000 and leap year calculations. Furthermore, Year 2000 compliant information technology shall accurately process date/time data if other information technology properly exchanges date/time data with it."

1.6 Widespread understanding of the Year 2000 (Y2K) challenge is an emerging phenomenon. While technologists understand the many intricacies in how computer software handles dates, explanations in the media and elsewhere tend to simplify the problem for brevity. Descriptions of the Y2K phenomenon are typically straightforward. Software commonly carries only the lower two digits for each year and, as a result, ignores century when manipulating dates. Once software begins mixing dates in both the 19xx and 20xx ranges, errors of varying degrees can result. In some cases, the software may fail immediately without completing its operation. In others, the software may continue to produce erroneous output, which can remain unnoticed.

1.7 There has been extensive examination of the Y2K challenge and ways to address it. These include but are not limited to:

- Essential Y2K program activities (inventory, impact assessment, etc.)
- Prioritizing, scheduling, and coordinating conversion projects
- Vendor management
- Sizing the Y2K effort
- Options for representing and manipulating date data
- Methods for converting existing applications
- Inventory and conversion tools

1.8 While examining various aspects of this subject is vital, the considerations discussed above are actually elements of the solution--not the problem. Technical experts tend to talk about what changes are needed in how software and hardware handles and stores date data. Program managers tend to talk about planning, estimation, and other deliverables. Higher-level managers tend to talk about the cost and the diversion of resources. Each person struggling with the Y2K challenge tends to focus on that part of the solution uppermost in his or her mind. The first step in addressing the Y2K challenge is to identify mission risks that include:

- Eliminating century risk. In reality, this risk can only be eliminated for a period of time--the longer the period, the greater the cost.
- Preserving current operations. Changes to achieve compliance should have minimal affect on the cost and efficiencies, as well as security of current operations.
- Minimizing conversion burdens. The Coast Guard must carefully manage resources commited to the Y2K effort.

1.9 The challenge in defining compliance is to enable implementation plans that address the business needs while satisfying the following considerations:

- Assure that all aspects of the Y2K challenge are adequately covered
- Encompass a full spectrum of cost-benefit-risk trade-offs
- Permit wide latitude in adopting technical and programmatic solutions
- Develop a standard process to test and verify compliance activities at all levels of operation

This process attempts to encourage solutions that minimize Y2K risk while minimizing the cost of Y2K conversion and maximizing the assurance for success for every Coast Guard unit.

1.10 Century compliance, however, is larger than the rollover to 2000. There are several other risk areas whose events are associated with the Y2K challenge. These events can occur not only at the rollover to 2000 but well before or well after. There are various elements of the Y2K technical issue as shown in examples of events below. It is important to understand that alternatives for addressing each element share considerable overlap. One of the best examples is Century Ambiguity.

1.11 Century Ambiguity is the most common element found in Y2K research. Software represents dates with a 1-digit or 2-digit year. When the software does not recognize that the dates are not all in the 19xx range, the results are undesirable. Examples:

- Data edits reject years in early 20xx as invalid.
- User interface does not allow 4-digit year to clarify century.
- Sorting leaves dates for 20xx and 19xx in jumbled order.
- Duration such as invoice aging is calculated incorrectly. The century is truncated or changed between entering and retrieving a date.
- Comparing a date in 19xx with a date in 20xx assumes both are in 19xx.

1.12 Examples of Century Ambiguity can appear in the following events:

- Bank ATM rejects an otherwise valid bank or credit card with an expiration date of "00".
- Lotus 1-2-3 accepts only 2-digit years, which it assumes to be in 19xx only.
- Itemized monthly bill lists transaction for Jan 1, 2000 through Jan 15, 2000 followed by Dec., 19, 1999 through Dec., 31, 1999.
- Invoice age calculated as ridiculously large number or as negative number, erroneously triggering overdue notices and staggering interest penalties
- Software stores dates in the 20xx range using DBMS but only passes 2-digit years to the product.
- Payroll-deduction calculations for years in 20xx incorrectly mistake the year as 19xx and fail to apply recent changes in tax laws.

1.13 Given these considerations, Y2K events can occur with timing as early as:

- First use of [credit] cards issued in 1995.
- First need to enter values later than 1999 (has already occurred; e.g. enlistment’s, officer commissions, etc).
- First monthly data processing in the year 2000.
- Calendar roll to January 2000.
- 1996 for systems with 5-year time horizon or really any time horizon beyond the last day of 1999.
- First quarter of the year 2000 (quarterly calculations extending beyond 1999).

II. Purpose and Scope

2.1 Purpose

The Coast Guard Year 2000 Management Plan focuses on Y2K resolution efforts throughout the Service. It provides guidance for inventorying systems, prioritizing systems, renovating or retiring systems, and monitoring progress. **Coast Guard unit commanders** are responsible for awareness, assessment, renovation, validation, implementation and contingency planning actions. The purpose is to (1) provide guidance for a Coast Guard-wide coordinated effort that ensures mission critical and non-mission critical systems or pieces of equipment are repaired in accordance with the Chapter VIII timeline and, (2) emphasize that contingency plans should be prepared to minimize the impact of any failures which may occur in any system that is not fully repaired.

2.2 Scope

This plan applies to all facets of the Coast Guard information infrastructure, all systems supported by information technology, their technical environment, and their communications devices, including but not limited to: automated business information systems, automated command and control systems, and weapon systems. Information technology support includes hardware inclusive of equipment containing embedded microprocessors, firmware, commercial off the shelf (COTS) and government off the shelf (GOTS) developed software, and data. Software includes COTS/GOTS packages, operating systems, third and fourth generation language compilers and interpreters, functional applications, system utilities, translators, and database management systems (DBMSs). Data includes databases, files, and other data storage structures and mechanisms, data and system interfaces and interchanges, Electronic Data Interchange (EDI) transaction sets and implementation conventions, and other messages or forms of data exchange.

It also applies to all electronic interfaces between the Coast Guard and external organizations including the Department of Transportation, Department of Defense, other federal, state, and local government agencies, the private sector, and other organizations required for the conduct of Coast Guard business.

III. Goal and Objectives

3.1 Year 2000 Goal and Objectives

The Coast Guard goal is to repair mission critical and non-mission critical systems and prepare contingency plans to minimize the impact of Y2K related problems. Objectives include:

- Minimize the adverse impact of Y2K date processing in all support systems.
- Define and share Coast Guard-wide, consistent strategies for finding and fixing Y2K problems, and testing solutions.
- Minimize duplication of effort for finding and fixing Y2K problems, and testing solutions.
- Minimize the impact of resource reallocation to support Y2K efforts.
- Minimize risk and cost in determining the appropriate Y2K solution for each system.
- Recognize the Y2K problem as an opportunity to retire legacy systems early.
- Identify, prioritize, and mobilize needed resources for system conversions and replacements.

IV. Management Strategy

4.1 Five Phase Strategy

The Coast Guard management strategy calls for centralized policy and Coast Guard-wide implementation using the five phase process, decentralized funding, and support by program and project management activities. This allows Coast Guard units the flexibility to address the problem as they see fit while also benefiting from best practices in a well-coordinated effort. Y2K efforts are underway and require parallel execution of portions of our five-phase solution process. The five phases are:

- Awareness. The first phase focuses on promoting Y2K awareness throughout the Coast Guard.
- Assessment. The second phase consists of system inventory and problem assessment.
- Renovation. The third phase constitutes systems replacement, retirement, or conducting repairs to ensure Y2K compliance.
- Validation. The fourth phase in which systems are tested for Y2K compliance and interoperability.
- Implementation. The fifth and final phase is systems deployment.

4.2 Unit Responsibilities

It is essential that units address all phases of the Y2K resolution process. The Coast Guard's Management Plan focuses on centralized management with decentralized execution. In addition to those responsibilities noted in Chapter VII of this document, **Coast Guard units are responsible** for their internal awareness, assessment, renovation, validation, and implementation actions. **Units are also responsible** for and should **reprogram base funding** to fix unique systems, not centrally supported that are impacted by Y2K problems. Units who find that the cost of their Y2K project exceeds available resources should document these costs and forward them through their chain to the appropriate headquarters program manager. Supplemental funds provided by congress may be available to meet some of these costs.

4.3 Information Sharing

To reduce duplication of effort and leverage Y2K experiences, information on Y2K problems, best practices, and lessons learned will be shared. The primary sharing media for this effort are the Y2K-related homepages on the Internet. This sharing of information will include sharing with other Coast Guard units, government agencies, and the private sector. See Appendix F for references and identification of web sites. Of particular interest see the USCG Operations Systems Center (OSC) web site for a significant listing of systems status.

4.4 Completion Target

The target for completing all Y2K repair, validation(i.e. testing), and implementation efforts is March 31, 1999, however every effort should be made to complete these phases as soon as possible. This will allow time to deal with unanticipated problems that may occur.

4.5 Resourcing

Fixing Y2K problems is the Coast Guard's top software and equipment resource priority. Units must therefore allocate/reallocate existing (base) resources accordingly, to fix Y2K problems. Elevating Y2K to this level priority will cause delays for some change request proposals and pre-planned software improvements. However, based on Congressional intent, units will postpone software enhancements and change request proposals until units have checked, analyzed, fixed, tested, and verified all computer systems, software, and electronic equipment for Y2K compliance. It doesn't make sense to improve software that hasn't been verified to work properly because of Y2K issues.

4.6 Prioritization

Priority of repair for mission critical and non-mission critical systems and equipment is:

- Safety
- Operations
- Administration
- Others

Mission critical systems must be repaired and Y2K certified first.

4.7 Coast Guard Standard Date Format

Where practical Coast Guard units shall use a four-digit contiguous year for the year portion of dates used for interfaces among systems and in all interagency information exchanges. Based on a recommendation from the federal CIO Council, OMB published policy adopting the four-digit-year date format as the standard for all data exchanges and systems interfaces in the federal government. Coast Guard policy requires the four-digit-year date format for systems interfaces and data exchanges within the Coast Guard to reduce the risk of re-infection of Y2K problems in our systems and databases, and to minimize the internal Coast Guard requirement for data bridges and translators.

In Electronic Commerce (EC)/Electronic Data Interchange (EDI) transactions, where other formats might be used, units will use four digit year representations when they are available, will use the century indicator ("CC") when it is available, or will use translators as necessary. The century indicator is the first two digits of a four-digit year (CCYY). An example of a standard calendar date in the CCYYMMDD format representing 12 April 1985 is 19850412. The year and month format will be CCYYMM, and the year format will be CCYY.

When system applications require the use of Julian (Ordinal) dates, the Ordinal day of the year is represented by three numeric digits. The first day of the calendar year will be 001 and subsequent days are numbered in ascending sequence. The year field will be represented by four digits; years are numbered in ascending order according to the Gregorian calendar. For example, the ordinal date representation of 12 April 1985 will be 1985102.

4.8 Coast Guard System Termination/Retirement/Elimination

Systems and electronic equipment that do not support the organizational missions or ones that could be combined into other systems are prime candidates for termination. Additionally, this is an opportunity to eliminate unnecessary systems from the inventory. System termination is the preferred solution to a Y2K problem if a risk assessment determines that it is a viable option. In some cases, the "window of vulnerability" (period during which dates will be improperly processed) of a system is small. A unit may decide that the system will not be used during that period, or that a temporary "workaround" solution can be applied during the "window" and removed afterward.

4.9 Replacement Alternatives

Coast Guard units will take advantage of Y2K-compliant Commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) solutions whenever practical to replace a system that has Y2K problems. When cost effective, another replacement alternative is to rapidly redevelop the system through rapid application development (RAD), rapid architect application development (RAAD), Business Process Reengineering (BPR), or object technologies and methodologies.

V. The Five Phase Management Process

5.0 The five phase management approach was developed by the Air Force and has since been adopted by GAO and other government agencies.

5.1. AWARENESS PHASE

The Coast Guard is aware of the Y2K problem. Its potential impact on Coast Guard missions requires continuous and ongoing awareness on the part of all members.

5.1.1 Define the Problem

The "Y2K problem" is the term used to describe the potential failure of information technology (IT) prior to, on, or after January 1, 2000. This potential exists because of the widespread practice of using two digits, not four, to represent the year in computer databases, software applications, and hardware microchips. Difficulties will arise in the Y2K when that year is 00 and our information technology will be unable to differentiate it from the year 1900.

5.1.2 Establish a Unit Y2K POC

The first step in attacking Y2K is to establish a unit level Y2K point of contact (POC). Although most major Y2K issues are at the Headquarters/Headquarters unit level, Y2K has the potential to affect every unit in the Coast Guard. Unit commanders must decide the level of resources and attention needed to protect themselves against Y2K problems which might affect their software programs and electronics for which they are directly responsible (unit developed, unit procured, HQ chartered responsibility, etc).

5.1.3 Identify Technical and Management Representatives

Another area critical to successful Y2K resolution is the identification of technical and management points of contact (POCs). This includes system managers, budgeting and resource personnel, legal representatives, senior management, support contractors and other external contacts, particularly of data exchange partners.

5.1.4 Desktop and Distributed Computing Systems

The Coast Guard has in recent years seen considerable increase in the utilization of desktop computing. The interfacing and data exchange between various computing systems must be addressed for Y2K problems to ensure proper data handling and conversion for the Y2K. Units down to the District level need to determine dependency links between internal and external systems, between core mission areas, processes and all data exchange entities, and provide for date and data format conversions where necessary. Data bridges and filters may provide some of the solutions for external links, but a validation process is necessary to ensure compliance. **UNISYS and MICROSOFT have provided appropriate certifications and software modifications to ensure that SWII and SWIII will be Y2K compliant.**

5.1.5 Coast Guard Contracts

Coast Guard contracts should follow the guidance provided in the Federal Acquisition Regulations, (FAR 39.106) that addresses Y2K compliance definitions and language.

The following principles and guidelines are consistent with the FAR and should be considered:

- The Coast Guard will only purchase Y2K-compliant computers, computer-related hardware and software, electronics, and equipment containing embedded microprocessor chips. Verbal assurances of vendors are unacceptable – contract language or other written documentation must provide evidence of compliance.
- The Coast Guard will use Y2K compliance language in contracts.
- The Coast Guard will issue stop work orders on all contracts for new products being purchased for products that fail to meet Y2K requirements or modify those contracts to correct Y2K deficiencies.
- Contracting offices will ask contractors to develop a Y2K compliance plan to upgrade their Y2K non-compliant products.

5.2 ASSESSMENT PHASE

This phase deals with those activities required to define the scope of the problem and set up the infrastructure necessary to solve it. The main deliverables from this phase are a complete inventory of computer systems, software applications, and electronics as well as the Project Plan for specific systems. Program sponsors have ultimate responsibility for the completeness of the inventory of systems and equipment under their purview. Although the focus is on mission critical systems, unit commanders must assess **all** systems and equipment important to their operations.

The primary purpose of the Assessment Phase is to gather and analyze information necessary to determine the size and scope of the problem. Only after the size and scope of the problem have been determined can an estimate of the cost in dollars and time be made.

Coast Guard units are to continually assess the impact of Year 2000 on their information technology hardware, software, and devices to include the Year 2000 impact caused by microchips.

5.2.1 Code Inventory

The Code Inventory involves locating all of the code that must be modified for the Year 2000. The volume and type of code will help determine the magnitude of the problem we have with software applications. Source code may be housed in a single repository, or the ownership and responsibility for maintenance may be decentralized and spread over the work force. Having the code in a centralized repository of some type, whether it's an elaborate vendor library package or a simple partitioned data set, is a big help in solving the Y2K problem.

All code must be inventoried and tracked and its relationship to other code determined. A total count of the lines of code (LOC) will assist in determining how many and what type of resources will be required to make the changes.

5.2.2 Collecting Survey Information

An assessment survey can be used to gather the necessary information about source code which is not contained in a central repository. The survey process seeks to collect system data in a standard format which can then be stored in a central database or spreadsheet. Collecting data such as system IDs, descriptions, units (language, platform, etc.), interfaces, owners/users/maintainers, and other relevant information is critical to any future efforts such as prioritizing and scheduling systems for renovation. Every system, even those currently selected for migration or retirement, must be inventoried.

The assessment survey should be distributed to all units. It will yield several important results:

- The survey provides the first indication of the level of effort that must be expended. When the results of the survey are summarized, a very rough cost estimate can be applied using $1.10 per executable line of code (ELOC) for administrative systems and $8.00 ELOC for command, control, communications, computers, and intelligence (C4I) systems. As assessments progress, more detailed estimates based on projected engineering costs, person-hours, and testing requirements should be used.

- The survey may also identify that common units exist which can be managed from the program manager level. The collection and distribution of information regarding in-house and vendor tools and services is a good example. Knowing which platforms and languages are most common will allow Y2K management to focus their efforts more precisely, such as in the evaluation of tools for analyzing systems written in specific languages.

- Finally, even if a central repository is in existence for the bulk of a unit's code, it is not uncommon, due to the proliferation of personal computers and local area networks, for individual departments to develop their own computer systems. The survey will help to identify these systems so that awareness, assessment, and renovation can be extended to them. Additionally a survey which is properly worded will generate input from those infrastructure systems (for example, environmental systems such as heating, security, and elevators) which must also be date tested.

5.2.3 Missing Source Code

Missing source code increases both the scope and cost of the project because it requires time and resources to develop both the functional and program specifications in order to rewrite the missing modules. Some units will discover, as they inventory their code, that there are some programs that are running for which the source code is no longer available. In some cases these programs have been running for years. Unfortunately the Y2K issue requires that every piece of code be examined to determine if any two digit date handling is involved. This may require the re-creation of all programs that have no source code.

There are several ways to address this issue. First, assess the impact of failure of this system and determine if there is already a replacement system in development. If there is no replacement, but the impact of failure is low, then the cost of rewriting or disassembling the object code may not be cost effective. Instead, plan for how to deal with the possible failure. However, if there is no replacement, but the impact of failure is high, then consider one of the following options:

- Assign the task to in-house programmers to have them either rewrite the code from the original specifications (if available) or disassemble the object code and try to re-create the source from that.

- Send the object code to a vendor who has the ability to re-create source through a combination of existing software tools and proprietary products. In both cases the final result should be source code that can be examined for Y2K compliance and is easily maintainable.

- Replace or terminate the system all together.

5.2.4 Mapping Source To Executables

A one-to-one mapping of source code to executable code should be made in order to determine that the source code in the inventory actually corresponds to the executable code running in production. There are several products on the market that allow you to map your source code to your executables. In general, they each produce a report that breaks all the load modules in a particular load library and displays pertinent information about the source code.

5.2.5 Vendor Software

All the operating system software and program products that surround the application software may need changing. The first step in this process is compiling a comprehensive list of vendor software used by your unit. With the exception of Coast Guard Standard Workstation II or III bundled software, software procured from vendors must be Y2K-compliant – stated in writing by the vendor. For systems and/or software that have been supplied by a specific program manager, units, through their chain of command Y2K POC, should contact that program manager for information concerning vendor software.

5.2.6 Contractor Maintained Software

Some units will also have to deal with the issue of contractor-maintained software. This class of software may need Y2K modifications. This may be done by the contractors. If outside the scope of the contract, and/or if time is not available for the contractors to perform the Y2K work outside of the Operations and Maintenance (O&M) work, units may need to contract a separate Y2K solution provider to resolve Y2K problems.

5.2.7 Embedded Microchips

Any piece of equipment that contains a microchip is at risk of having a Y2K problem. When assessing your environment for potential Y2K issues, stretch your imagination to identify equipment that contains the elusive, date-processing embedded microchip. For example, VCRs and video cameras contain date-related chips to program in a date to record an event. Elevators, climate control systems, pacemakers, automobile computers, Global Positioning System (GPS) receivers, private branch exchanges (PBXs), and engineering control machines may be overlooked when conducting a Y2K assessment inventory. If in doubt, check with the program sponsor or original equipment manufacturer (OEM), through your chain of command Y2K POC, for a Y2K certification listing.

5.2.8 Pilots

As the source code issues are resolved and an inventory is established, consideration of budget and scheduling issues is required. System owners, users, designers, and developers cannot assume any system is Y2K-compliant until it has been certified. Estimating the amount of time required to actually make a module/program Y2K compliant is difficult. It will depend upon the complexity of the program, whether documentation exists, the skill level of the programmer, and the familiarity of the programmer with the program in question. One process is to conduct a pilot, maintaining careful records of the time required to modify the code for Y2K compliance. The code selected should be representative of the bulk of the inventory. The time spent can then be extrapolated over the total inventory, to produce a work year estimate. If more than one pilot is conducted, using modules of varying complexity, the work year estimate gained can then be weighted across the inventory, depending upon the percentage of code by complexity.

Another reason for conducting a pilot is to determine if a Y2K automated tool would be beneficial. Piloting an application without the aid of a tool and comparing the results against the same or similar code modified with the assistance of one or more of the tools available will show whether savings can be achieved. Another reason for conducting a pilot would be to determine which approach should be used when changing the code. Several applications could be piloted using different approaches and the results compared to see which is the most cost effective.

Be aware, the use of pilots can consume a considerable amount of time and resources.

5.2.9 Identify Technical Issues

At this point all other technical issues that could affect the project should be identified. Consider them as you develop your cost estimates, risk management plans, and contingency plans. Following are some examples of issues that will affect the cost of solving the Y2K problem:

Forms -- pre-printed and computer generated
On-site vs. off-site contractor resources (off-site adds costs)
Screen issues (2 or 4 position years)
Library clean-up
Format of dates on inputs and outputs
Standardized date routines
Databases and archives

5.2.10 Estimating System Costs

There are a number of factors that will influence the cost of making systems Y2K compliant in addition to modifying software. They include building the test environment, buying automated tools and services, adding hardware, upgrading operating system software and commercial products, buying embedded microchip upgrades, etc. The Department of Defense has developed a checklist for “estimating system costs for the Y2K,” which includes those additional items that must be considered. The checklist will indicate those areas where costs should be adjusted because of your specific environment.
See Appendix A, "Year 2000 Cost Factors Checklist".

If the unit has obtained an accurate means of developing a cost estimate, use it.

The Coast Guard is using a combination of cost metrics developed by the Gartner Group and MITRE Corporation, costs estimated by Commandant (G-SI/Y2K), and costs from internal unit estimates. For C4I systems, the cost algorithm is $8.00 per executable line of code (ELOC). For all other systems, $1.10 per ELOC is applied. These metrics allow the Coast Guard to estimate costs for system evaluation through validation and implementation.

As stated earlier, as assessments progress, systems managers must provide more detailed estimates based on projected engineering costs, person-hours (including in-house as well as contractor) and testing requirements as they become available. Also Coast Guard units may base cost estimates on actual fixes. Units must identify the methodology used.

5.2.11 Tools

Testing tools, re-engineering tools, and other types of tools may be necessary at various times. There are two main types of tools that are being marketed to deal specifically with the Y2K problem. The first type consists of those tools that change the system date for an individual batch job. This allows units to determine how specific programs or systems will handle processing for any chosen future date.

Tools that help locate dates and related fields, and in some cases change those dates in the source code, comprise the second category. These tools usually provide a pre-defined set of fields such as “century", “year”, “yy”, etc., to which fields known to be dates in specific systems can be added. The tools then examine the code for occurrences of these fields and also track the flow of dates as they move from field to field. The output produced by these tools helps pinpoint those areas in the code that must be examined most closely.

There is no silver bullet, i.e., a tool that will find all date fields in the code and make the necessary changes. No one has developed such a product and most experienced programmers do not believe one is possible given the nearly infinite possibilities for storing and manipulating dates and because of individual programming methodologies.

The list of Year 2000 vendors and tools on the Defense Information Systems Agency (DISA) homepage will provide more information on those vendors who have tools, (See Appendix F for references and web sites).

5.2.12 Procurements

Procurements may be required at any point during the Y2K project. An overall procurement strategy should be developed during this phase and refined in later phases. Tools to be used during the renovation phase should be bought here. This includes tools that help with the analysis, tools that allow system date manipulation, and common date routines. Once the assessment has been completed, it will be possible to determine whether additional manpower will be needed, or whether the project can be completed with existing resources. Units who find that the cost of their Y2K project exceeds available resources should document these costs and forward them through their chain to the appropriate headquarters program manager. Supplemental funds provided by congress may be available to meet some of these costs.

5.2.13 Risk Analysis/Prioritization

Only after an inventory of embedded microchip systems and source code has been completed and a rough estimate of the size and scope of the problem within a unit has been obtained, can decisions regarding priorities be made. When the data gathering portion of the assessment phase has been completed and resource estimates are available, some units will realize that it is impossible to do everything.

If work cannot be completed in time, units need to undertake a risk analysis of their systems and assign each a priority based on its mission criticality. Units that get a late start on the problem and/or do not have the necessary infrastructure in place may find themselves with not enough time to convert and test even critical systems. Contingency plans are required.

5.2.14 Develop Validation Approach

The Assessment Phase should include the development of a strategy and a validation schedule. The schedule should indicate the general time frames for the validation of all systems and should consider hardware concerns such as availability of processing cycles and storage, along with human resource issues.

Future date testing should be done in isolation with no possibility of corrupting or destroying production files. If the resources need to be available in late 1998, the appropriate reprogramming and procurement actions should already have started. Validation should be completed as soon as possible; the OMB validation completion date is 31 January 1999.

5.2.15 Electronic Data Interchanges

Electronic Data Interchanges (interfaces) involve the sending and receiving of data between Services and/or Defense Agencies or external Coast Guard vendors, etc. The National Institute of Standards and Technology issued a FIPS (Federal Information Processing Standard) Publication Change notice on March 25, 1996, which stated: “for purposes of electronic data interchange in any recorded form among U.S. Government agencies”, NIST highly recommends that four-digit year elements be used. The year should encompass a two-digit century that precedes, and is contiguous with, a two-digit year-of-century. In addition, optional two-digit year time elements specified in ANSI x3.30-1985 (R1991) “should not be used for the purposes of any data interchange among U.S. Government agencies.”

Electronic Data Interchanges are critical in the Y2K effort because they have the potential to introduce and/or propagate errors from one Coast Guard unit to another. Where 4-digit dates cannot be coordinated in a timely fashion, bridges can be written to accommodate the transition period. Bridges receive information in one format, modify it, and put it out in another format, such as receiving the year in a 2-digit format, adding century information through the use of an algorithm, and writing the output with a 4-digit year. Where the bridge must accommodate data from a number of sources, it can be table driven, to accept the input in either format, based upon an entry in the table, but output the data in a fixed 4-digit year format.

When bridges are written as separate modules/steps, they can be more easily removed when the sending/receiving programs become fully compliant. Existing documentation should be updated to reflect the insertion of bridges or the use of filters, sliding windows and algorithms.

5.2.16 Developing a Plan

The unit's plan, started during the Awareness Phase, should be completed as the last step in the Assessment Phase. The plan should show, at a minimum, milestones within each phase including the start date and release date for each phase, the major steps to be taken in converting and testing the code and establishing the necessary infrastructure, and the resources required to accomplish these tasks. Beyond these basics, the project plan should include plans for dealing with data interfaces, and contingency plans as noted below.

The most challenging aspect of the plan will probably be the scheduling of each system. This will require tremendous coordination within the unit and between units that exchange data. Where possible, the goal should be to convert related systems simultaneously to reduce the number of bridge programs that must be written and maintained.

5.2.17 Develop contingency plans

Unlike routine system development or maintenance efforts where schedule slippages are non-fatal--and common--the Y2K program must be completed on time, or Y2K failures may affect our readiness. Units should develop realistic contingency plans, including the development and activation of manual or contract procedures, to ensure the continuity of their core processes should Y2K compliance efforts fail.

The general position widely held is that some percentage of critical systems will fail. Although we don't know how or when these failures might affect us, we must develop contingencies to continue core mission responsibilities and to support our Team Coast Guard members and their families.

Contingency plans should be updated at each phase.

5.3 RENOVATION

The Renovation Phase involves making and documenting software and hardware changes, developing replacement systems, and eliminating systems. Renovation involves conversion of an existing application; replacement deals with the development/purchase of a new application; elimination focuses on the retirement or decommissioning of an existing application or system unit. In all three cases, the process must also consider the complex interdependencies among applications, hardware platforms, databases, and the internal and external interfaces.

The processes that may be in the Renovation Phase are:

5.3.1 Conversion

Convert selected applications, databases, archives, and related system units. In converting application systems, consider changes in operating systems, compilers, utilities, domain-specific program products, and commercial database management systems.

NOTE: It may be prohibited to alter historical or other archived data for legal reasons. A common sense approach is recommended. If historical or archived data is sensitive due to the likelihood of its use in future litigation, personnel claims against the government, or similar functions, consult Commandant (G-SII) or your servicing legal department prior to undertaking conversion.

5.3.2 Develop Data Bridges and Filters

Ensure that all internal and external data sources meet the Y2K-compliant date standards of the converted or replaced systems. Develop bridges to convert non-conforming data. Filters may be used to edit out non-conforming data.

5.3.3 Replacement

Replacement deals with the development of a new application or expansion of an existing application. Selected applications, platforms, database management systems, operating systems, compilers, utilities, and COTS software which are not in compliance may be replaced. Ensure that replacement products are Y2K compliant, including their ability to properly handle the leap year adjustments. Contract specialists and legal staff are to review contracts and warranties.

5.3.4 Document Code and System Changes

All changes to the information systems and their units should be made using configuration management procedures to ensure that changes are adequately documented and coordinated throughout the agency. Equally important is the need for each agency to assess dependencies and to communicate all changes to the information systems to internal and external users.

5.3.5 Unit, Integration, and System Tests

Schedules for unit, integration, and system tests following the conversion of individual application and software modules should be completed. Coordinate scheduling with other units to ensure that all system units, including data bridges or filters, are available for testing. Systems will likely require additional modifications during validation phase. Test schedules must include time for regression testing.

5.3.6 Elimination

Selected applications, platforms, database management systems, operating systems, utilities, COTS software, and equipment containing microprocessors may be eliminated if the cost of repairing them is not justified by their value to Coast Guard missions. As appropriate, prepare to terminate applications, platforms, database management systems, operating systems, utilities, COTS software, and equipment containing microprocessors not being replaced.

5.3.7 Communicate Changes

It is necessary that all changes to Coast Guard information systems and units, and specifically all changes to date formats for data exchanged with other systems or external organizations be communicated to affected Coast Guard units and outside organizations. It is necessary to communicate those changes to the user community. Document changes through the configuration management process.

5.3.8 Track the Conversion and Replacement

Track the conversion and replacement projects and collect and use project metrics to manage cost and schedule. This record should include a record of the unit’s Y2K costs.

5.3.9 Share Information

Ensure that project staffs understand the need to collect and disseminate information on lessons learned and best practices. Develop dissemination strategy and tools, such as web sites, newsletters, etc., or request Commandant (G-SI/Y2K) to disseminate the information (See Appendix F for references and web sites).

5.4 VALIDATION

Coast Guard units will need an extensive period of time to adequately validate and test converted or replaced systems for Y2K compliance. Gartner Group estimates the testing and validation process could consume 50-60% of the time needed to complete a Y2K project. The length of the validation and test phase and its cost are driven by the size of the application and the complexity inherent in the Y2K problem. Coast Guard units must not only test Y2K compliance of individual applications, but also the complex interactions between scores of converted or replaced computer platforms, operating systems, utilities, applications, databases, and interfaces. In some instances, units may not be able to shut down their production systems for testing, and may have to operate parallel systems implemented on a Y2K test facility. Three processes are involved in Y2K validation:

Validation Process - All converted or replaced system units must be thoroughly validated and tested to (1) validate Y2K compliance, (2) uncover errors introduced during the Renovation Phase, and (3) verify operational readiness. The testing should account for application compliance, database interdependencies, and interfaces. The testing should take place in a realistic test environment. Each Coast Guard unit should assess their testing procedures, facilities, and tools to ensure that all converted system units meet quality standards and are ultimately certified Y2K compliant (See Appendix-B, "Guidelines For Y2K Testing" and Appendix-C "Year 2000 Compliance Checklist").

Independent Validation Process - Coast Guard units should conduct an Independent Validation if there has been significant changes to any system. An Independent Validation is a second validation effort to increase confidence that full remediation work has been done on a Coast Guard system by one of three methods: (1) CG System Experts must witness and validate the testing process and test scripts during each validation evolution, (2) CG System Experts recall remediators to step through crucial portions of Y2K Test criteria with them, enough to determine that critical inputs/outputs are consistent and that critical Y2K remediation for that platform has been accomplished, (not a full IV&V), or (3) Employ an Independent Contractor to accomplish that listed within (2) above. Most cost effective and desirable of the three methods of Independent Validation outlined above is (1).

CIO Verification Process - Commandant (G-SI), the Chief Information Officer will designate various Systems to undergo an examination of the the methods used to conduct the two processes above. In some instances a mission critical system may undergo a combination of both the IV And CIO Verification. The CIO Verification will be managed by the Year 2000 Project Staff, Commandant G-SI/Y2K.

5.4.1 Develop and Document Tests, Plans, and Schedules

For each converted/replaced application or system unit, units should develop and document test and compliance plans and schedules. Units should establish a compliance validation process, and should define, collect, and use test metrics to manage the testing and validation process.

5.4.2 Develop a Strategy for Managing the Testing of Contractor-converted Systems

In some instances, a Coast Guard unit will contract for the conversion of selected systems and the system units. The contract conversion must be closely managed to ensure that the contractor follows the Coast Guard's Y2K conversion date standards. In addition, the unit must ensure that the contractor-converted systems are adequately tested and certified.

5.4.3 Implement a Y2K Test Facility

The requirement for a Y2K test facility must be determined by individual units. Where such facilities are implemented, testing the converted or replaced systems and the system units for Y2K compliance will likely require an isolated test facility capable of simulating Y2K requirements. The test facility should provide for large test databases and multiple versions of the application software.

5.4.4 Implement Automated Test Tools and Test Scripts

Units may take advantage of computer-aided software testing tools, and test scripts have the potential to significantly reduce the testing and validation burden. Test management tools may help in the preparation and management of test data, in the automation of the comparison of test results, in scheduling and incident tracking, and in managing test documentation.

5.4.5 Perform Testing

Use selected testing techniques to ensure that the converted or replaced systems and accompanying units are functionally correct and Y2K compliant.

5.4.6 Initiate Acceptance Testing

Acceptance testing is the final stage of the multiphase testing and validation process. During this phase, the entire information system -- including data interfaces -- is tested with operational data. Testing should be done at a Y2K test facility or in a separate environment with duplicate databases to avoid risk to the production systems and the potential contamination of data.

5.4.7 Complete Acceptance Testing

In general, formal testing uncovers about 80-90 percent of software errors, with the remaining 10-20 percent of errors discovered during operations. Acceptance testing should be completed no later than January 31, 1999, to allow sufficient time for the correction of software errors discovered following implementation.

5.4.8 Update Contingency Plans

Unlike routine system development or maintenance efforts where schedule slippages are common but not fatal, the Y2K program must be completed on time. Units should update contingency plans, including the development and activation of manual or contract procedures, to ensure the continuity of their core processes.

5.5 IMPLEMENTATION

Implementation of Y2K compliant systems and their units requires extensive integration and acceptance testing to ensure that all converted or replaced system units perform adequately in a heterogeneous operating environment. Because of the scope and complexity of the Y2K conversion changes, integration, acceptance, and implementation may be lengthy and costly.

Once converted and subsequently tested, renovated applications and system units must be implemented. Since not all system units will be converted or replaced simultaneously, units may be expected to operate in a heterogeneous computing environment composed of a mix of Y2K renovated and non-renovated applications and system units. The reintegration of the Y2K renovated applications into the unit's environment must be carefully coordinated to account for system interdependencies. Parallel processing--where the old and the converted systems are run concurrently--may be needed to reduce risk.

5.5.1 Define Transition Environment and Procedures

The transition from the current environment to Y2K renovated systems will be difficult and complex. First, some key parts of the unit's systems -- Y2K renovated databases, operating systems, utilities, and COTS products -- may not be available until late 1998 or early 1999. Second, external data suppliers may not plan to complete their conversion and testing until 1999. Third, the testing, validation, and correction processes may take much of 1998 and may extend into 1999. Fourth, replacement systems may not be ready for testing until late 1999. As a result, units may be forced to operate, at least for a time, parallel systems and databases.

5.5.2 Develop Implementation Schedule

The Y2K implementation schedule must deal with uncertainties common to all large system development efforts, and should indicate all major milestones and the critical path for the completion of the Y2K program.

5.5.3 Resolve Data Exchange Issues, Interagency Concerns, and Ensure the following:

- Internal and external data exchange entities are identified, and contacted. The Coast Guard's target date for completing this task was March 1, 1998.
- A plan is agreed upon for data bridges and filters to handle non-conforming data. A written Memorandum of Agreement is completed with the data exchange partner.
- Contingency plans and procedures are in place if data exchange is interrupted.
- Contingency plans, developed during the Assessment Phase are updated and in place if invalid data is received from an external source.
- The validation process is in place for incoming external data.

5.5.4 Deal with Database and Archive Conversion

Because the conversion of large databases from 2 - digit to 4 - digit year fields is a time consuming effort, units should consider off-site conversion alternatives.

5.5.5 Post Implementation Considerations

Implementation completes the phases delineated in the Y2K Management Plan. However, a period of close monitoring of systems should follow through the year 2000 to ensure that compliant systems work reliably. When January 2000 arrives, any problem that surfaces may be attributed to the Y2K problem. Coast Guard units should be aware of the necessity of monitoring after the change in the calendar year and after the Year 2000 leap year day. The possible expenditure of additional funds to correct any unforeseen problems that may occur should be part of a contingency plan. All verified problems related to the Y2K should be documented. Additionally, other key dates throughout the year should be watched carefully for Y2K related problems. See Appendix B, "Guidelines For Y2K Testing", for testing guidance associated with some post-January 1st, 2000 dates.

VI. Performance Indicators

6.1 The Y2K computing challenge comes with ultimate performance measures - January 1, 2000 for many systems and October 1, 1999 (the beginning of FY2000) for most federal systems. Does your system work prior to, on, or after that date? In the meantime the Coast Guard will measure progress towards a Y2K solution using a variety of indicators. Current performance indicators for Y2K problem solution are (but are not limited to):

- The number of systems in each phase of the five phase management process; ultimate success is a system certified **Y2K compliant** and **fully implemented**.
- The number of systems scheduled for elimination (not being replaced).
- Total recorded and estimated costs.

6.2 Effective September 30, 1997, the Coast Guard Y2K performance indicators, in addition to the indicators above, include:

- The total number of systems in Coast Guard's inventory;
- The number of systems eliminated;
- The number of systems impacted by the Year 2000 problem;
- The total number of interfaces;
- The number of interfaces impacted by the Year 2000 problem;
- The number of interfaces renovated or coordinated;
- The number of systems renovated, and certified Y2K compliant;
- Remaining systems to be renovated;
- The number of Y2K renovated systems implemented.

6.3 Commandant (G-SI/Y2K) will include the performance indicators from sections 6.1 and 6.2 when measuring progress.

VII. Responsibilities

Commandant (G-SI), as the Coast Guard's Chief Information Officer, has the overall management responsibility for Y2K resolution within the Coast Guard. To assist in the Y2K cross-functional issue resolution process, the CIO created the Coast Guard Y2K Working Group.

7.1 Coast Guard Year 2000 Working Group

The coordinator of Y2K information for the Coast Guard Commandant, (G-SI/Y2K), chairs the Coast Guard Y2K Working Group. Each representative on the Working Group will investigate Y2K and cross-functional issues, provide recommendations, and identify and share lessons learned. Primary membership on the Working Group includes representatives from all HQ Assistant Commandants. Other members will be added as needed. Commandant (G-SI/Y2K) will send periodic Y2K information updates to Working Group members via e-mail.

7.2 Commandant (G-SI)

Commandant (G-SI), as the Coast Guard's CIO has the following responsibilities:

- Establish Coast Guard-wide strategies and policy guidance for addressing the Y2K problem.
- Set Coast Guard-wide goals for completing each phase of Coast Guard Year 2000 activities.
- Oversee Coast Guard-wide Y2K planning and implementation of Year 2000 activities.
- Establish Y2K reporting requirements
- Oversee Coast Guard Y2K contingency planning and business continuity activities

7.3 Commandant (G-SI/Y2K)

Commandant (G-SI/Y2K), as the Coast Guard's Y2K Project Office, has the following responsibilities:

- Support the Coast Guard CIO in executing Coast Guard-wide Year 2000 initiatives.
- Develop and maintain the Year 2000 Data Base (CG-Y2KDB) which will contain an inventory repository and associated Y2K information on computer systems, software applications, and equipment using embedded microchips throughout the Coast Guard.
- Facilitate the flow of Y2K information throughout the Coast Guard. Maintain a list of other government or industry Y2K information sites listing of tools and Y2K solution providers available to units which can assist in resolving the Y2K problems.
- Respond to Congressional, OMB, DOT, other agency, and internal Y2K data calls.

- Hold Coast Guard wide awareness activities. Carry out site visits as appropriate.
- Maintain liaison on Y2K issues with other government agencies, and representatives of Coast Guard partner industries.
- Record overall Coast Guard Y2K costs

7.4 Assistant Commandants

Assistant Commandants have the following responsibilities:

- Implementation of this plan or equivalent.
- Prepare and execute a Y2K oversight program for systems under their purview.
- Identify and prioritize mission critical and non-mission critical systems consistent with approved management strategy.
- Discontinue or replace old applications systems, determined to be too costly to repair.
- Monitor the execution of Y2K corrections for systems within their functional areas.
- Document and obtain Coast Guard-internal and external system interface agreements in the form of Memorandums of Agreement (MOAs) or equivalent.
- Make resource decisions and develop strategies for systems with Y2K problems within their functional area(s). Identify budget shortfalls and include them in budget submissions and reprogramming actions.
- Purchase and/or develop only Y2K-compliant systems.
- Include Y2K compliance language in all new contracts and contract modifications as appropriate.
- Identify a Y2K POC responsible for all Y2K questions and actions within the Program Manager’s functional area(s).
- Develop individual Year 2000 plans for each mission critical and non-mission critical system, with milestones geared to programmatic requirements.
- Provide responses to data calls from Commandant (G-SI/Y2K) as required. This includes input to the Coast Guard's Quarterly OMB Y2K report to the Department.

7.5 Coast Guard Area, MLC, District Commanders and Units

- As appropriate, discontinue, replace, or renovate non Y2K-compliant, locally-developed or purchased software applications.
- Replace non Y2K-compliant, locally-purchased equipment and electronics.
- Purchase and/or develop only Y2K-compliant systems, software, and equipment.
- Appoint a Y2K POC for all Y2K actions within the command or unit.

7.6 (G-OCI) Office of Intelligence

- Coordinate Coast Guard intelligence activities in addressing the Year 2000 challenge.

VIII. Timeline

<u>8.1 The timeline has five phases</u>. The end dates represent target completion dates for each phase. Some phases must overlap in order to complete all actions no later than March 31, 1999. The timeline presented here is consistent with the Office of Management and Budget's timeline. However, it is expected that Coast Guard units will complete each phase as quickly and as thoroughly as possible.

This aggressive schedule is necessary due to the limiting factor in the Y2K project – Time. Units should target these dates for completing exit criteria and beginning and completing phases.

Contingency planning begins in the Assessment Phase and is updated in the Renovation, Validation and Implementation Phases.

<u>8.2 Phase I (Awareness)</u> - Awareness, education, and initial organization and planning take place. Awareness will continue throughout all Year 2000 phases and activities.

Target Completion Date for Initial Awareness: <u>December 1996</u>

-- Exit Criteria:
- --- Phase I plan completed and distributed.
- --- Corporate strategies developed and submitted to the Coast Guard CIO.
- --- Y2K POCs for all major organizations identified and educated.
- --- System users and owners identified and educated.
- --- Key Coast Guard and industry POCs contacted.
- --- Phase II strategy developed, documented, and distributed.

<u>8.3 Phase II (Assessment)</u> - Scope of Y2K impact is identified and system level analyses take place.

Target Completion Date: <u>July 1997</u>

-- Exit Criteria:
- --- Phase II plan completed and distributed.
- --- 100% inventory of all systems keyed into CGY2KDB (Target: July 31, 1997).
- --- Phase III strategy developed, documented, and distributed.
- --- 100% of systems to be replaced, renovated, retired, identified and confirmed (Target Completion Date: July 31, 1997).
- --- 100% of systems analyzed for Y2K compliance.
- --- Y2K resource strategy and plan developed and completed.
- --- 100% of systems requiring renovation prioritized and scheduled for Phase III.
- --- Risk management and contingency strategy developed, documented, and distributed.

<u>8.4 Phase III (Renovation</u>) - Required system "fixes" are accomplished.

Target Completion Date: <u>September 30, 1998</u>

-- Exit Criteria:
- --- Phase III plan completed and distributed.
- --- Successful renovation of all scheduled systems.
- --- Phase IV plan completed and distributed.
- --- Phase V strategy developed and completed.
- --- Risk management and contingency strategy updated.

8.5 Phase IV (Validation) - Systems are confirmed as Y2K renovated through assorted testing and compliance processes.
Target Completion Date: January 31, 1999
-- Exit Criteria:
--- Unit, integration, and system testing completed, systems certified.
--- Acceptance testing and certification completed.
--- Independent verification of system repair.

8.6 Phase V (Implementation) - Systems are fully operational after being certified in Phase IV.
Target Completion Date: March 31, 1999
-- Exit Criteria:
--- Risk management and contingency strategy updated and distributed.
--- Systems successfully integrated and operational.
--- Independent verification of system repair and successful implementation provided.

APPENDIX A: YEAR 2000 COST FACTORS CHECKLIST

This appendix is for internal unit use only. Its purpose is to assist units in estimating/tracking system hardware and software application Y2K costs. It is the responsibility of all units to estimate and record costs associated with Y2K through all phases, from assessment through implementation. The following method may be used to develop cost estimates: Go through the checklist and mark all factors that apply to your system. Next, develop relative determinations of how these factors affect your cost estimate. Finally, apply these relative determinations to refine your cost estimate. Additionally, you can use the factors checked to consider in developing your test, contingency, and risk plans.

A.1 **Application Software**:

____ Size: Number of executable lines of code (ELOC)

____ Age: Older code tends to be less structured and thus harder to understand

____ Complexity: Relative intricacy/understandability of the business rules

____ Documentation: Degree of documentation available and its understandability

____ Programmer: Familiarity with the program code. Level of skill/competency/expertise

____ Source Code: Availability

____ Date- "Intensiveness": Relative number of date related calculations/comparisons

____ Embedded Dates: Frequency of date use as part of data element or in data element codes

____ Date Formats Used: Consistency within the system of a standard date format

____ Year 2000 Strategy (Field expansion/procedural code/sliding window): Different strategies to achieve Year 2000 "compliance" have different costs.

____ Language: Some coding in older languages such as COBOL 68 may have to be upgraded and changed. These older languages may not be compatible with COTS tools and programmers may not be available.

A.2 Hardware/System Software: Year 2000 compliance of each of the units of the technical environment is required. [Often only the most recent version of a product will be Year 2000 compliant.]

____ Operating System

____ Major Subsystems: Sometimes subsystems have different technical environment units.

____ Database Management System (DBMS)

____ Compilers/cross-assemblers (if available - sometimes they don't exist)

____ Teleprocessing (TP) monitors

____ Homegrown/locally developed software that is used in conjunction with the system

____ Workstation Software: Consider the quantity needed.

____ Workstation BIOS (handles the "system clock function"): 60-80% of PC BIOSs are not Year 2000 compliant -- most are soldered to the "motherboard, " some are reprogrammable, some are "socketed" and some can be replaced.

____ Programmer: Familiarity with the hardware and operating system; level of skill/competency/expertise

____ Programmer System Software (utilities and development tools): To support making changes to the software

____ Capacity/Usage Level: Making a system Year 2000 compliant may increase storage requirements or even CPU requirements and necessitate purchase of a larger computer.

____ Embedded Software (microchips/circuit cards; e.g., PBXs, security system access control, cash registers): They may be directly or indirectly related to a system, and may not be Year 2000 compliant. The availability of compliant hardware or the cost of developing, and the quantity required must be considered.

____ Communications: Telecommunications hardware and software upon which the system depends must be considered.

____ Network Timestamps (LAN/WAN network, clock time): Upon which the system is dependent

A.3 Database/Files:

____ Number of date-related data elements

____ Amount available

A.4 Year 2000 Tool Support:

____ Availability: Many languages and/or technical environments do not have Year 2000 COTS tools so tools must be developed in-house or specifically contracted for development.

____ Quality

A.5 External Interfaces/Middleware:

____ Data Sources: Must be evaluated and "bridges" planned as required.

____ Data Outputs: Must be evaluated and "bridges" planned as required.

____ EDI Transaction Sets: System may generate some Electronic Data Interchange(EDI) transactions or get input from EDI transactions which require "bridges."

____ Reports: Systems may generate paper reports which need to be modified.

____ Screens: Systems may have screens used by users which require modification.

A.6 System Plans:

____ Planned Major Upgrade: May be an opportunity for Year 2000 compliance work/system modification at the same time to reduce costs.

____ Termination: System may be eliminated before a Year 2000 problem occurs.

____ Replacement: System is planned for COTS replacement or reengineering before impacted by the Year 2000.

A.7 Miscellaneous System-Related Information:

____ Sort Routine Year 2000 compliance

____ Backup Routine Year 2000 compliance

____ Archival Routine Year 2000 compliance

____ System Criticality/Priority: Really not required for cost estimate, but a good time to record this critical planning information.

____ Risk Analysis (if system fails): Really not required for cost estimate, but a good time to record this critical planning information. Consequences of system failure, including cost, must be considered.

____ Risk Analysis (if system is not made Year 2000 compliant): Many systems only have a small "window of vulnerability" during which not being able to process Year 2000 properly occurs. Consideration must be given to whether or not this "window" is acceptable; i.e., the system won't be used during that period, or a "workaround" will be established for that period; e.g., manual processing.

____ Contingency and Continuity of Operations Planning (significant costs may be connected with making some backup systems contingency-ready.)

A.8 Year 2000 Management:

____ Project Management

____ Configuration Management

____ Change Management

____ Contract(or) Management

____ Year 2000 Emergency Reaction Team

A.9 Year 2000 Testing:

____ Establishing Test Environment

____ Unit Testing

____ Integrated Testing

____ Year 2000 Simulation Testing: Can sometimes require mirror of production environment. Might not be possible until technical environment is made Year 2000 compliant.

APPENDIX B: YEAR 2000 TESTING GUIDANCE

B.1 Criteria for Century Compliance

This testing process for century compliance requires that the following areas satisfy four date-related criteria:

- Hardware and firmware
- Operating System Software
- Application and Middleware Software
- Network Systems (Hardware, software, and firmware)
- Databases (including engines, repositories and archive systems)

The above elements meeting all four criteria can be considered "Century-Compliant." The four date-related criteria to apply to these elements are:

Extended Semantics

a. In general, specific values for a date field are reserved for special interpretation. The most common example is interpreting "99" in a 2-digit year field as an indefinite end date, i.e., "does not expire." Another is embedding a date value in a non-date data element.

b. Some software may erroneously process a transaction with a valid end date in 1999, such as not terminating an expired software license or failing to age back-up tapes for recycling, as scratch tapes.

c. This will occur on various days after December 31, 1998.

Calendar Errors

a. Errors typically include failing to treat 2000 as a leap year and converting incorrectly between date representations.

b. Logic sensitive to day-of-week will be one day off after February 28, 2000. Calculating day of week for all dates following this will be incorrect.

c. This will occur the first time input data contains February 29, 2000. This will occur on this leap day.

Date Overflow

a. Many software products represent dates internally as a base date/time plus an offset in days, seconds, or microseconds since that base date/time. Hardware integers holding the offset value can overflow past the maximum corresponding date; an event which may lead to undefined behaviors.

b. Value for date can revert to a date near the base date/time, to a negative value, or crash the software because of an illegal operation.

c. Happened in the 1980s on certain Tandem hosts. Could happen again at any time to any product depending on how product stores dates.

Inconsistent Semantics

a. At interface between systems, software on each side assumes semantics of data passed. Software must make same century assumptions about 2-digit years.

b. Software on one side assumes all dates in 19xx. Software on other side assumes years 51-99 are 19xx, and 00-50 is 20xx.

c. This could happen in 1996 for software that stores date values 5 or more years into the future.

B.2 Criteria for Y2K Testing

Each criterion serves as a high-level requirement for operating system and application software and/or firmware. The following discussion elaborates on each:

Criterion	Description
General integrity	No value for current date will cause interruptions in normal operation.
Date integrity	All manipulations of calendar-related data (dates, duration, days of week, etc.) will produce desired results for all valid date values within the application domain
Explicit century	Date elements in interfaces and data storage permit specifying century to eliminate date ambiguity.
Implicit century	For any date element represented without century, the correct century is unambiguous for all manipulations involving that element.

Table B-1 Four Y2K Testing Criteria

General Integrity-- As a system date advances normally on a host processor, each date rollover must not lead either the host process or any software executing there to erroneous processing. The best recognized, high-risk, date change is the rollover to 2000.

Date Integrity-- This criterion primarily covers the correctness of manipulations of date data (see Calculation Manipulations below). These manipulations need to be reliable only over the range of dates that an application is expected to handle. For example, sales-order processing may handle dates from 5 years in the past to one year in the future. In contrast, an employee database may store dates of birth from early in the 20th century to planned retirement dates well into the 21st century.

Explicit Century--This criterion essentially requires the capability to store explicit values for century. For example, third-party products that can use a 4-digit year in all date data elements stored and passed across each interface (including the user interface) would satisfy this criterion. A base-and-offset representation of dates that covers all centuries of interest would also satisfy this criterion. Whether this capability should be used to eliminate century ambiguity is part of the last criterion.

Implicit Century--This last criterion requires that, if the century is not explicitly provided, its value can be correctly inferred with 100% accuracy from the value of date provided. For example, the range of values for an "invoice date" would very rarely span more than 10 years. Because the century can always be guessed correctly for an invoice date with a 2-digit year, this date data element would satisfy this criterion. Note that this criterion permits cost-risk trade-offs that minimize changes to existing date formats.

Although the four criteria fully define century compliance, compliance represents a balance between cost and risk rather than an absolute "yardstick." Such a balance will vary with each organization according to its business needs and its technological base. Consequently, an organization will require a greater level of detail to interpret how these criteria apply to the organization. An interpretation documented and distributed can assure that "century compliance" is interpreted consistently across the organization.

Calculation Manipulations:

Arithmetic
- Calculate the duration between two dates
- Calculate date based on starting date and duration
- Calculate day of week, day within year, week within year

Branching
- Compare two dates

Format
- Convert between date representation (YMD, Julian, etc.)
- Reference same data address with different variables

Data Storage
- Storing and retrieving
- Sorting and merging
- Searching
- Indexing on disk file or database table
- Moving data within primary memory

Extended Semantics
- "99" as special value for year
- "99.365" as special value for Julian date
- "00" as special value for year

Table B-1 above contains an interpretation of the criteria as an example. It is important to note the need to clearly identify the calendar by name, the specific date ranges for compliance relevant to the organization, reasonable latitude in date format, and situations under which implicit century values will be tolerated. Also note that certain exceptions are included to support important options for optimal cost/risk trade-off.

Standardizing the format for date data is an important part of century compliance. The examples in the following list uses the ANSI X3.30 standard for date representation. For systems with interfaces to foreign entities or other U.S. government agencies, the equivalent ISO and FIPS standards may be used instead. There are two very important considerations which require conformance to industry standards:

Limitations in standards. None of the three standards (ANSI, ISO, and FIPS) for date representation **mandates** a 4-digit year for all calendar data. For example, conformance to ANSI X3.30 does not eliminate century ambiguity from all date variables and interfaces. Instead, conformance simply reduces the variety of formats occurring within software and data.

Accommodating conflicts. While trying to conform to ANSI X3.30, some applications may need to satisfy other standards or conventions for date representation.

General Integrity
No value for current date will cause interruptions in normal operation.

- All software on all platforms will function correctly for all values of system date between 1985-01-01 and 2035-12-31.

- Of special interest are the following dates and the ability to roll over to the correct next date: 1998-12-31, 1999-06-30, 1999-07-01, 1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-28, 2000-02-29, 2000-03-01, 2000-12-31, 2001-01-01, 2027-12-31.

Date Integrity
All manipulations of calendar-related data (dates, duration, days of week, etc.) will produce desired results for all valid date values within the application domain.

- All date values and calculations are based on the Gregorian calendar as defined in Encyclopædia Britannica, 15th edition, 1994, p. 430.

- Computing assets must correctly handle all representation and manipulation of dates with values between 1900-01-01 and 2050-12-31. Especially important is that all years in this 150-year range divisible by 4 are leap years except 1900.

- All software developed should initialize all date elements with either all zeros (0000-00-00) or null values. Null values are defined for each application by the development facilities, such as language compiler. A null-value feature is strongly recommended in third-party-software selection.

- All developed software must not contain literals or constants for dates unless required to capture specific business rules such as calculations of payroll deductions.

- All developed software must not use special date values as logical flags, such as "99" as year to mean "no end date" or "00" to mean "does not apply."

Exceptions:

- Valid date ranges in existing developed or existing third-party software may start with the oldest date value in the application's archived data rather than with 1900-01-01 when there is no business need to support earlier dates.

Explicit Century

Date elements in interfaces and data storage permit specifying century to eliminate date ambiguity.

- All developed and third-party software must permit the use of date formats which explicitly specify century in all date data stored or transmitted. The format of these date elements must be CCYYMMDD or CCYYJJJ as specified by ANSI X3.30-1985 (R1991) unless superseded by another application-specific standard or convention.
- In storing or transmitting date data, some applications must conform to domain-specific standards, contractual agreements, or APIs to necessary third-party products whose date formats must supersede ANSI X3.30 as appropriate within the application.
- Third-party products must permit formatting data with explicit century in the user interface.
- All developed applications using third-party products must always explicitly supply century and never rely on those products default value for century.

Exceptions:

- For date data formatted for a user interface, it is acceptable to use punctuation (slash, hyphen, period, and comma) within a formatted date, to spell out or abbreviate the name of the month, or to reorder year-month-day to serve customs among the end-users.
- DataBase Management Systems (DBMS) that cannot store date in conformance with SQL standards but do store century explicitly (such as DD-MM-CCYY), are acceptable.
- Default values for century are permitted only when supplied by data-entry aids and the end-user can verify the defaulted value before committing the data.

<u>Implicit Century</u>

For any date element represented without century, the correct century is unambiguous for all manipulations involving that element.

- Century must be explicit in all date data stored or transmitted unless the correct century can be inferred with 100% accuracy based on the value for date. Explicit century is preferred where practical.
- Developed and third party software may imply century in the user interface in the format, YYMMDD or YYJJJ (as specified by ANSI X3.30).
- In storing or transmitting date data, some applications must conform to domain-specific standards whose requirements for dates may supersede ANSI X3.30 as appropriate within the application.

Exceptions:

- For date data formatted for a user interface, it is acceptable to use punctuation such as slash within a formatted date, to spell out or abbreviate the name of the month, or to reorder year-month-day to serve customs among the end-users.

<u>B.3 Applying The Criteria</u>

The above information can be applied throughout the Coast Guard. Century compliance should become both a standard in normal unit operations and a fundamental specification for Y2K specific efforts.

Units should adopt the below criteria as standards in day-to-day business:

- **a.** <u>Software development</u>. Developments of new software, both in-process and planned, should immediately adopt a standard for century compliance.
- **b.** <u>Third-party product selection</u>. Selection of third-party software and intelligent hardware products shall be governed by the compliance criteria and explicitly specified in contract language.
- **c.** <u>Service-vendor selection</u>. Selection of vendors for data processing, marketing data, and other services must demonstrate compliance with these or equivalent Y2K criteria.

Compliance criteria are the most important concept in a Y2K program. All other goals, decisions, and plans must revolve around a common understanding of compliance. The compliance criteria should be applied to the following tasks of a unit's Y2K program:

For Software:

1. Designing inventory. When inventorying software assets, the compliance criteria can help ensure that the appropriate data is collected. For example, if third-party products are used in custom application software, then knowledge of non-compliant third-party products will help identify non-compliant applications.

2. Impact analysis. As each computing asset is examined for compliance, a clear statement of century compliance is essential to ensure that all elements of the Y2K issue are identified. For example, third-party products should be evaluated against certification testing based on the criteria above.

3. Software conversion. When deciding how best to convert an application to achieve century compliance, the compliance criteria should guide those decisions. For example, each date field in permanent storage must be examined to determine what modifications to the data and/or program logic are needed to achieve century compliance.

4. Software testing. During Y2K conversion projects, converted software must be tested to verify that it is now century-compliant and that none of the functionality has been impaired during the conversion. The compliance criteria should guide the design of these tests.

For Hardware:

1. Developing inventory. When inventorying hardware assets, the compliance criteria can help ensure that the appropriate data is collected. For example, going to vendors/suppliers for Y2K certification of particular pieces of equipment can identify whether complex changes need to occur or just operational testing for compliance. Knowledge of non-compliant products will help identify non-compliant systems and therefore focus work on what has to be fixed and tested.

2. Impact analysis. As each computing asset is examined for compliance, a clear understanding of how this piece of hardware fits into the system is essential to ensure that all elements of the Y2K issue are identified. For example, hardware products should be evaluated against the generic certification testing criteria herein despite its perceived individual priority or importance.

3. Hardware or firmware conversion. Vendor criteria should help guide decisions about converting a piece of hardware or firmware to achieve century compliance. Use the enclosed criteria to question and test vendor fixes and alterations. For example, a date field in permanent storage (firmware or ROM) must be examined to determine what modifications are needed to achieve century compliance. Identifying this to the vendor and seeking the compliance change is much better than just asking if the equipment is Y2K compliant.

4. System and component testing. During Y2K conversion projects, converted or fixed hardware and firmware must be tested to verify that it is now century-compliant and that none of the functionality has been impaired during the conversion. The compliance criteria should guide the design of these tests. **Don t accept a paper certificate in lieu of real life, production testing of any equipment regarding Y2K compliancy!**

B.4 Impact Analysis and Testing

Testing against the century-compliance criteria falls into three categories:

General-- Some aspects of century compliance are independent of technology and application domain. For example, January 1, 2000 follows December 31, 1999 for all hardware and software.

Technology-specific-- Some aspects of century compliance depend on the technology and the behaviors and features of that technology. For example, not all software performs all the date manipulations previously listed.

Domain-specific-- Some aspects of century compliance depend on the requirements of a specific application. For example, the valid ranges for dates to be tested can vary with each application.

The following sections provide a high-level guide for testing by category. Testing should be organized in this manner to permit reusing test procedures among different applications and systems. The categories below can be applied to any or all of five system areas (hardware, software applications, operating systems, databases, and networks) and serve as a guide to begin testing processes.

General

Current date
- a. Direct set, power-up roll-over
- b. Re-initialize from cold start
- c. Full date ranges

Calendar accuracy
- a. Days of week in 2000 and 2001
- b. Leap year in 2000
- c. 366 days in 2000

Century ambiguity
- a. High-risk values (1999-09-30, 1999-12-31, 1900-01-01, 2000-01-01, 2000-02-29)
- b. Ambiguous century in user interface
- c. Electronic interfaces (API such as O/S call for system date, etc.)

Technology-specific

Current date
- a. Power-down continuity
- b. "System date" versus "current date"

Date representation
- a. Overflow of base-and-offset representation
- b. Standards for Gregorian formats
- c. Century capability in storage and interfaces

Date manipulation
- a. Date arithmetic
- b. Conversion between representations
- c. Sorting, searching, indexing
- d. Century designation available in storage and interfaces

Domain-specific

Century ambiguity
- a. Accuracy in inferring century for each field in storage, user interface, and other interfaces.

Date representation
- a. Industry standards and contract requirements
- b. Human-factors requirements
- c. Design and coding standards

Date manipulation
- a. Access to archived data
- b. Manipulation of archived data
- c. Extended semantics

Century compliance as previously described, is based on the philosophy that the Y2K challenge must be viewed as not only a technical but also a mission-ready issue. The spirit of compliance is to eliminate technical risks in computing assets through cost-effective and operationally sound solutions. Therefore, if there are several conversion options that have an acceptable cost and will support operational schedules, it is recommended not simply to select the least-cost among them but the one option that most closely attains all of the following objectives:

Explicit century-- All custom and third-party software and data explicitly represent century in all date data elements external at software interfaces, internal to the software logic, and stored both on-line and off-line.

Variations in representation-- The number of variations in syntax for date data elements is less than five. This includes all internal representation (except representation internal to third-party products), representation at all interfaces (not counting applicable industry standards), and representation in archived data.

Variations in user interface-- The number of variations in date syntax in the visual user interface of any application are no more than four. The term, "user interface", includes screens, windows, character-oriented dialogues, LED read-outs, paper reports, on-line reports, generated member mailings, user-accessible logs, etc.

Event horizon-- For century-compliant software, the next Y2K event is at least **50 years** away.

Common routines-- All applications in the same source language share the same date routines for all date manipulations.
Semantic extensions-- All extensions to date semantics are eliminated.

Summary
The Y2K challenge is a very urgent matter. Responding to the challenge without some analytical discipline in the front-end could impede achieving timely and cost-effective century compliance. Each unit should begin its Y2K program by selecting clearly articulated criteria applicable to its hardware, software, networks, and databases for century compliance and then use these criteria to guide the planning, execution and documentation of all activities.

Consider the Y2K pyramid of testing. Going from the known to the unknown, you can build on the elements that can already be verified and base your correction and testing procedures for the next steps on these building blocks. Testing a software program on a computer that has a non-compliant BIOS puts the value of the software test into question, especially if the application pulls system clock information from the computer. Begin testing the environmental and hardware bases first and proceed through the pyramid. Overlap as many of these pyramid functions as possible. Remember that testing begins when Y2K coding changes are completed for the first application program.

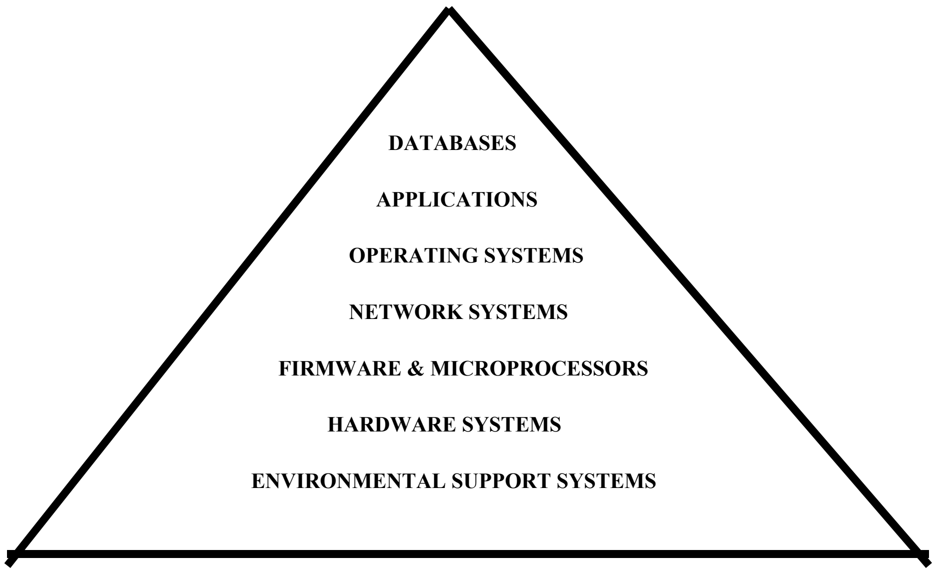

Table B-2 Y2K Testing Pyramid

B.5 Guidelines For Y2K Testing

I. Guidelines for Y2K Testing of a Hardware Computing Device

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

I.(a) General Integrity

__ System date on machine can be correctly set to high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)

__ Machine correctly re-initializes from a cold start on high-risk dates:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)

__ System date rolls over correctly on high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

I.(b) Date Integrity

__ Do date-sensitive services for system dates behave correctly in 19xx and 20xx?

- Date-sensitive includes specific dates, days of week, and duration.
- Services such as call-back, 800 routing, voice-mail operator

__ The system date correctly uses time-stamps for high-risk dates.

- High risk dates: (1998-12-31, 1999-01-01, 1999-09-30, 1999-10-01, 2000-01-01, 2000-02-28, 2000-02-29, 2000-03-01)
- Time-stamps are used correctly in call-data records, tracking and performance reports, and data saved on hard-disks

I.(c) Explicit Century

__ Can date-sensitive user interfaces* be set and used successfully for dates in 20xx? (Y/N)__________

If no, was this corrected? (Y/N)______. How was this corrected successfully? _________________

__ Are there 1- digit and/or 2-digit year interfaces used? (Y/N)______

If yes, were these covered in your corrective repair and testing actions successfully? (Y/N)______.

I.(d) Implicit Century

__ Are there any date-fields in any machine user interface* that contain a year field? (Y/N)______

If yes, was this brought into Y2K compliance? (Y/N)______. How was this corrected successfully?

__ Is there any way the interface* could misinterpret current entries for dates in 20xx? (Y/N)______

If yes, was this corrected? (Y/N)______. How was this corrected successfully? _________________

__ Is there a vendor or mfg. certificate of Y2K compliance for this piece of hardware? (Y/N)______

If yes, did you test the machine for the above anomalies? (Y/N)______

* = User Interfaces refer to any display panel, digital or analog display, meter, etc that allows interactive communication with date or date/time.

II. Guidelines for Y2K Testing of an Operating System

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

II.(a) General Integrity

__ System date can be set to high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)

__ Re-initialize from cold start on high-risk dates:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)

__System date rolls over correctly on the following high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

__ System date rolls over correctly in both powered-up and powered-down states.

II.(b) Date integrity

__ Do date-sensitive functions for system dates behave correctly in 19xx and 20xx? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

(Date-sensitive refers to functions whose behavior can start or stop on a specified date, on a specified day of the week, or after a specified elapsed time. Functions such as job scheduling, accounting, admin functions, any interactive command with dates in parameters or switches (especially disk and tape commands and utilities), setting or obtaining system date through system call, reporting system data with dates in it.)

__ System date appears correctly when used in time-stamps for high-risk dates

- High risk dates: (1998-12-31, 1999-01-01, 1999-09-30, 1999-10-01, 2000-01-01, 2000-02-28, 2000-02-29, 2000-03-01)

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ System date appears correctly when used in time stamps for queue entries, volume directories, inter-process communication, off-host communication.

II.(c) Explicit Century

__ Can century be explicitly entered or displayed in date-sensitive functions? (Y/N)______.

__ Are there functions or system calls that cannot permit an explicit century and whose format is governed by an industry standard or customer requirement? (Y/N)______.

If yes, was this corrected? (Y/N)______. What was the standard used, if necessary? ______________

How was this corrected successfully? ____
____
____
____

II.(d) Implicit Century

__ Are all century dates, in any date value, in any date-sensitive function, correct and operational for dates between 1985-01-01 and 2050-12-01? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

III. Guidelines for Y2K Testing of a Compiler, Assembler, or Interpreter

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

III.(a) General Integrity

__ Does the language provide a function to obtain or set system date on the host or through a time service? (Y/N)?

If yes, was this correct? (Y/N) ______. If yes how was this corrected successfully________________
____
____
____

__ Function returns the correct value for system date for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)

__ System date rolls over correctly on the following high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

III.(b) Date Integrity

__ Does the language support a data type for date values in the following range? (Y/N)______
(1900-01-01 to 2050-12-31)

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Do language routines treat 2000 as a leap year and 1900 as a non-leap year? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ The date arithmetic correctly calculates differences between dates, add dates and duration, and computes date of week.

__ Do the language routines correctly convert between dates all possible date format representations (YMD to Julian to base-and-offset)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does the language correctly compare dates? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does the language work with a library of date routines? (Y/N)______.

If yes, does the library perform date compares correctly? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does the system include sort/merge utility? (Y/N)______.

If yes, does sorting or merging on a date field produce correct sequence across dates in 19xx and 20xx? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

III.(c) Explicit Century

__ Date data types provide for correct and explicit values with the new century.

__ Are there functions or system calls which cannot permit an explicit century, and whose format is governed by an industry standard or customer requirement? (Y/N)______.

If yes, was this corrected (Y/N)______ and what was the standard used? ________________________
____

How was this corrected successfully? ____
____
____
____

III.(d) Implicit Century

__ Date data type supports formats without explicit century.

__ Value for century not explicitly set is assumed in: ____
(e.g., date comparisons, arithmetic, permanent storage, etc.)

__ Is century correctly inferred in all cases where century is not explicit? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Are all century dates, in any date value, in any date-sensitive function, correct and operational for dates between 1985-01-01 and 2050-12-01? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

IV. Guidelines for Y2K Testing of a Database Management System (DBMS)

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

IV.(a) General Integrity

__ Does the language/DB Protocol provide a function to obtain the system date through the host platform___, operating system___, the DB engine___, or through a time service___?

__ Does this function return the correct value for system date for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Database generated dates roll over correctly on high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

IV.(b) Date Integrity

__ Does the DB engine or DB language support a data type for date values in the range 1900-01-01 to 2050-12-31? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the DB engine or DB language and/or its routines treat 2000 as a leap year and 1900 as a non-leap year? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? _________________
____
____
____

__ The date arithmetic correctly calculates differences between dates, add dates and duration, and computes the correct date of week.

__ Do the DB engine or DB language routines correctly convert between date representations (YMD to Julian to base-and-offset internal)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the language correctly compare dates? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does a key index that includes a date field, produce correct sequence across dates in 19xx and 20xx? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the DBMS or DB engine retrieve dates accurately for values in the range 1900-01-01 to 2050-12-31? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

IV.(c) Explicit Century

__ Does date data type permit setting explicit values for century? (Y/N)______.

__ Do retrieval functions permit formatting dates with explicit century? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the DBMS or DB engine retrieve dates accurately for values in the range 1900-01-01 to 2050-12-31? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

IV.(d) Implicit Century

__ Date data types in the DB engine and/or DB language generate data types without explicit century. (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

V. Guidelines for Testing Y2K Compliance of a Local-Area Network (LAN)

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

General Integrity

__ Does client network software obtain correct system date from the host for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does server network software obtain correct system date from the host for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does network-administration software obtain correct system date from the host for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Network administration software and/or network server returns the correct value for system date after the system date rolls over on high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

__ Does LAN software accept and set time stamps correctly in the range of:
(1985-01-01 to 2035-12-31)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

Date integrity

__ Does admin software support expiration dates for user accounts, passwords? (Y/N)______.

If yes, was this also tested and shown to be corrected across the century? (Y/N)______.

If no, how was this corrected successfully? ____
____
____
____

__ Does calendar logic treat 2000 as a leap year and 1900 and a non-leap year? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ The date arithmetic correctly calculates differences between dates, adds dates and duration, and computes the date of week.

__ Does the LAN server or network operating system language routines correctly convert between date representations (YMD to Julian to base-and-offset internal)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the language correctly compare dates? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Do the network-admin software and API-to-network software accurately accept and retrieve all date fields in the range of values, 1900-01-01 to 2050-12-31? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

Explicit Century

__ Do time stamps provide for explicit values of century? (Y/N)______.

__ Does the user interface in network administration software permit explicit entry and display of century in all date fields? Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does the API-to-network software permit explicit entry and retrieval of century in all date fields? Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

Implicit Century

__ For time stamps without explicit century, is a value for century inferred correctly for all time stamp manipulations? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ For date fields in user interface of admin software, which value for century is assumed if not entered by the user? _________

__ For date fields in the API, which value for century is assumed if the application does not automatically set it? _________

VI. Guidelines for Y2K Testing of a Custom Software Application

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

General Integrity

__ Does language provide a function to obtain the system date on the host or through a time service? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does this function return the correct value for system date for high-risk dates:
(1999-09-09, 1999-09-30, 1999-10-01, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

__ Does this function return the correct value for system date after the system date rolls over on these high-risk dates? (Y/N)_____.
(1998-12-31 -> 1999-01-01)
(1999-09-30 -> 1999-10-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

__ Are third-party products embedded in this application? (Y/N)______.

__ Are all these products century-compliant? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? _____________
____
____
____

__ Does the application code ignore values for explicit century in the system date at any point in the program logic? (Y/N)______.

If yes, how is it handled? ____
____
____
____

<u>Date Integrity</u>

__ Does the programming language correctly support a data type for date values in the range of (1900-01-01 to 2050-12-31)? (Y/N)______

__ Does the application make a correct leap-year calculation? (Y/N)______

__ Do these calculations treat 2000 as a leap year and 1900 as a non-leap year? (Y/N)______

__ Does the date arithmetic correctly calculate duration (differences) between dates, add dates and duration, and compute date of week? (Y/N)______

__ Does the application correctly convert date values from one representation to another (for example, YMD to Julian to base-and-offset internal)? (Y/N)______.

__ Does software correctly convert between date representations according to the Gregorian calendar? (Y/N)______

If no, were all the above corrected? (Y/N)______. How were they corrected successfully? _________
____
____
____

__ Does the application compare dates in any of its branching logic or calculation of Boolean values? (Y/N)______

__ Do all these comparisons produce correct results for all combinations of values with the expected ranges for dates? (Y/N)______.

If no, were all the above corrected? (Y/N)______. How were they corrected successfully? __
____
____
____

__ Does the application include searching, sorting, merging, or indexing on internal tables, linked lists, or other data structures based on date variables? (Y/N)______

__ Do these operations perform correctly for all possible values for date in the key variables? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ____________
____
____
____

__ Does a key index, which includes a date field, produce correct sequencing across dates in 19xx and 20xx? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the application represent date in any variable as an offset from a base date/time? (Y/N)______

__ What is the maximum value for the date in this representation? ________________________

__ What is the minimum value for this date in this representation? ________________________

__ Does the expected range of values for each variable using this date representation fall within these extremes? (Y/N)______.

If no, were all variables corrected? (Y/N)______. How were they corrected successfully?
____
____
____

__ Does the application use application assigned values for the date, and pass it from one variable to another? (Y/N)______

__ Is the century portion of the value truncated during any assignment? (Y/N)______

__ Is the value in the target variable eventually used in a date manipulation that requires explicit century for correct results? (Y/N)______

If no, was this corrected? (Y/N)______. How were they corrected successfully? ____
____
____
____

__ Does the application use language features that map a data address to more than one variable (such as REDEFINEs in COBOL or COMMONs in FORTRAN)? (Y/N)______

__ In all aliases for the same data space, does any variable ignore or truncate values for explicit century in the date value? (Y/N)_____

__ Are the variable or truncated date values used correctly? (Y/N)______

If no, was this corrected? (Y/N)______. How was this corrected successfully?
____
____
____

__ Does the application store and retrieve dates accurately for values in the range of (1900-01-01 to 2050-12-31)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the application use sort/merge utilities to order file contents on date fields or use indexed file structures keyed on date fields? (Y/N)______

__ Is the order correct for all values of date in the range 1900-01-01 to 2050-12-31? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ____________
____
____
____

__ Does the application rely on primary or alternate indices on a structured database for search, insert, update, or delete in which any key contains a date field? (Y/N)______

__ Will the index order be correct for all values for date in the range of (1900-01-01 to 2050-12-31)? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ____________

____
____
____

<u>Explicit Century</u>

__ Does the application use a language, toolkit, and/or application generator that permits explicit century in the date data types? (Y/N)______.

If so, were values for century in variables of these types correctly derived from external input **<u>or</u>** correctly derived within the software logic? (Y/N) ______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ____________

____
____
____

__ Does the application use a DBMS or other layered (or horizontal) software product for data persistence to store and retrieve date variables? (Y/N)______

If so, can these products support explicit values for century in any date variable stored and retrieved? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? ____________

____
____
____

__ Does the application have external interfaces (I/O, APIs, external subprogram calls, IPCs, library routines, HMI) which contains a date variable with explicit century? (Y/N)______

__ Does the software ignore, truncate, or write over the century value in any such variable as it flows through the program logic to any other external interface? (Y/N)______

__ In any such flow, could any logic alter the value for century in any manner inconsistent with generalized manipulations based on the Gregorian calendar? (Y/N)______

__ Do all representations of date with explicit century both internal to the application and in all interfaces satisfy the criteria for century compliance? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully?

____
____
____

Implicit Century

__ Does the application use a language, toolkit, and/or application generator (including GUI builders) which permits date representation without an explicit century in the date data types? (Y/N)______ .

If so, is century derived for any manipulations or for passing a date value across any interface or for permanent storage? (Y/N)______ .

If so, is value for century correct for all possible values of date that each such variable can hold? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully?

____
____
____

__ Does the application use constant values for date or portions of date (i.e., day, month, or year). (Y/N)______.

If so, for any constant which is a full date value or value for year, is century explicit in the value? (Y/N)______.

If no, is it century compliant? (Y/N)______. If not, How will this be corrected successfully?

____
____
____

__ Do all manipulations using each constant value directly or indirectly (that is, carried via variables to other operations in the program logic), produce the correct results for all possible values for such date variables? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Does the application use any application-program interface (API), such as in-line SQL or IMS DML, which passes date variables? (Y/N)______.

If so, for any date value supplied across this interface, does the receiving software provide a default or derived value of century?(Y/N)______

__ Are the rules for derivation on both sides of the interface consistent with each other with possible values for date in their respective fields? (Y/N)______

If no, was this corrected? (Y/N)______. How was this corrected successfully?
____
____
____

__ Does the application correctly support user interfaces containing date fields without explicit century? (Y/N)______

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

__ Do all representations of date both internal to the application and in all interfaces satisfy the criteria for century compliance? (Y/N)______.

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

VII. Guidelines for Y2K Testing of Personal Computers

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

General Integrity

__ Was the PC made (not bought) after 1996? (Y/N)______.

If no, check the BIOS by interrupting the boot-up procedure after the memory test with a F1, F8, F10, Escape (Esc), or other interrupts specified by the manufacturer.

__ Can the system date on the machine be set successfully to high-risk dates of:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)? (Y/N)______

If no, you may need an updated BIOS. (Seek BIOS updates from the manufacturer's WEB site or phone the company's customer service department). If the BIOS is Y2K compliant or you've installed a new BIOS, continue with the following:

__ Re-initialize from cold start on high-risk dates:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)

__ Make sure the system date rolls over correctly on high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

Date Integrity

__ Do date-sensitive services (desktop functions) for the system date behave correctly in 19xx and 20xx? (Y/N)______. Date-sensitive includes specific dates, days of week, and duration. Services such as call-back, 800 routing, voice-mail operator.

__ System date is correctly used in time-stamps for high-risk dates such as:

- High risk dates of (1998-12-31, 1999-01-01, 1999-12-31 , 2000-01-01, 2000-02-28, 2000-02-29, 2000-03-01).
- Time-stamps in call-data records, track and performance reports, as well as data saved on the hard-disk.

Explicit Century

__ Can date-sensitive services be set and used successfully for dates in 20xx? (Y/N)______

If no, was this corrected? (Y/N)______. How was this corrected successfully? __________________
____
____
____

Implicit Century

__ Are there any date-fields in the operating system of the PC that contain a year? (Y/N)______

If yes, did the vendor certify this to be compliant for Y2K? (Y/N)______.

If compliance was indicated with issues, how were the issues corrected successfully? _____________
____
____
____

__ Are there any other desktop admin devices that might misinterpret dates in 20xx? (Y/N)______

If yes, what device was effected? (Y/N)______. How was this corrected successfully? ___________
____
____
____

VIII. Guidelines for Y2K Testing of Commercial Telephone Switches

(Check the **blanks__ that** apply and have been completed. Attach supporting documentation if applicable. Read each question although some may not apply. Check off those that do in the box to the left and add information as requested.)

General Integrity

__ System date on machine can be set to high-risk dates of:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)

__ Re-initialize from cold start on high-risk dates of:
(1999-09-09, 1999-12-31, 2000-01-01, 2000-02-29)

__ System date rolls over correctly on high-risk dates:
(1998-12-31 -> 1999-01-01)
(1999-12-31 -> 2000-01-01)
(2000-02-28 -> 2000-02-29)
(2000-02-29 -> 2000-03-01)

Date Integrity

__ Do date-sensitive services for system dates behave correctly in 19xx and 20xx? (Y/N)______.
(Date-sensitive includes specific dates, days of week, and duration. Services such as call-back, 800 routing, or voice-mail operator.)

__ System date correctly used in time-stamps for high-risk dates of:

- (1998-12-31, 1999-01-01, 1999-12-31, 2000-01-01, 2000-02-28, 2000-02-29, 2000-03-01)
- Time stamps in call-data records, track and performance reports, data saved on hard-disk

Explicit Century

__ Can date-sensitive services be set and used successfully for dates in 20xx? (Y/N)______

If no, was this corrected? (Y/N)______. How was this corrected successfully? ___________________
____
____
____

Implicit Century

__ Are there any date-fields in any interface that contain a year? (Y/N)______

If yes, was this corrected? (Y/N)______. How was this corrected successfully? __________________

____

____

____

__ Could either side of the interface misinterpret these for dates in 20xx? (Y/N)______

If yes, was this corrected? (Y/N)______. How was this corrected successfully? __________________

____

____

____

APPENDIX C: YEAR 2000 COMPLIANCE CHECKLIST

The purpose of this checklist is to aid unit Y2K managers in ensuring their systems are compliant for the Year 2000. Make sure the following testing and certification process is used for all of the developed, gratis, licensed, and purchased software, hardware, and firmware used in your unit's operation, development, maintenance, support, and testing activities. Locally reproduce this appendix and complete the checklist for each mission critical and non-mission critical system at your unit. Keep the completed checklists with your system records.

1. <u>System Identification</u>

a.	Name of system	
b.	Defense Integration Support Tools (DIST)	
c.	Operational date of system (current or a future date)	
d.	Planned or actual replacement date of system (retirement or discontinuation qualifies as replacement)	
e.	Is there a contingency plan for this system. Under what conditions will it be invoked?	
f.	What is the Priority rating of the system? (Mission Critical or Non-Mission Critical)	
g.	System Type (Either Hardware, Application Software, Operating System Software, Network, or Database)	

2. Year 2000 Considerations

Each system has its own window of time, before and after the present date, in which it functions. Planning and scheduling systems work with dates that are weeks, months, and sometimes years in the future. Likewise, trend analysis systems and billing systems regularly reference dates in the past. For your system, and its window of time, please verify its ability to successfully process or handle data containing dates with no adverse effect on the system's functionality and with no impact on the customer or end user beyond adjustment to approved changes in procedures and/or data formats.

2.		Yes	No	N/A	Verified
a.	Dates in 20th century (1900s)				
b.	Dates in 21st century (2000s)				
c.	Dates across century boundary (mix 1900s and 2000s)				
d.	Crosses 1999 to 2000 successfully				

3. Software Date Usage

Have you verified performance (and corrected if necessary):

3.		Yes	No	N/A	Verified
a.	Dates embedded as parts of other fields				
b.	Dates used as part of a sort key				
c.	Usage of values in date fields for special purposes that are not dates (e.g. using 9999 or 99 to mean "never expire")				
d.	Date dependent activation/deactivation of: passwords, accounts, commercial licenses				
e.	Date representation in the operating system's file system (creation dates and modification dates of files and directories)				
f.	Date dependent audit information				
g.	Date dependencies in encryption/decryption algorithms				
h.	Date dependent random number generators				
i.	Date dependencies in firmware				
j.	Personal Computer BIOS and RTC does not reset the year to 1980 or 1984 on reboots after 31 December 1999 *(corrections by operating system utilities allowed)*				

4. Leap Year

System accurately recognizes and processes Year 2000 as a leap year.

4		Yes	No	N/A	Verified
a.	February 29, 2000 is recognized as a valid date				
b.	Julian date 00060 is recognized as February 29, 2000				
c.	Julian date 00366 is recognized as December 31, 2000				
d.	Arithmetic operations recognize Year 2000 has 366 days				

5. Usage of Dates Internally

Internal application usage of dates and date fields must be clear and unambiguous in the context of the systems that use them.

5.		Yes	No	N/A	Verified
a.	Display of dates is clear and unambiguous (the ability to correctly determine to which century a date belongs either by explicit display, i.e. 4-digit year, or system or user inference)				
b.	Printing of dates is clear and unambiguous				
c.	Input of dates is clear and unambiguous				
d.	Storage of dates is clear and unambiguous				

6. External System Interfaces

External interactions are identified and validated to correctly function for all dates.

6.		Yes	No	N/A	Verified
a.	Interaction between this system and any other external time source, if existing, has been verified for correct operation. For example, the GPS system is sometimes used as a time source. Many GPS receivers cannot correctly deal with the rollover of the GPS 10-bit epoch counter that will occur at midnight, 21 August 1999. GPS receivers also deal with an 8-bit Almanac Week counter that has a 256-week rollover span.				
b.	You and the responsible organization for each interface have negotiated an agreement dealing with Year 2000 issues. For each interface that exchanges date data, you and the responsible organizations have discussed and verified that you have implemented consistent Year 2000 corrections that will correctly work for date data passed between your systems				
c.	The corrected Interface has been documented.				

7. Software or Database Date Field Type

Describe the type of date fields used by the system, in either software or databases.

7.		Yes	No	N/A	Verified
a.	Does the system use 4 digit year data-fields?				
b.	Does the system use 2 digit year data-fields?				
c.	If 2 digit, does the system use a century logic technique to correctly infer the century?				
d.	At what date will the century logic fix fail?				
e.	Are there any internal data types for dates? If yes, what is the range of dates that the date field can represent? Minimum Date:__________ Maximum Date:__________				

8. Year 2000 Testing Information

Please, provide the following information with regard to testing the application for Year 2000 compliance:

8.		Narrative Answer
a.	Testing Organization	
b.	Name of Test Team Leader	
c.	Date that Year 2000 compliance testing was completed	
d.	How was Year 2000 compliance determined? (Certified by vendor or contractor, tested in-house, inspected but not tested, etc.)	

8.		Yes	No	N/A
e.	For Software or Operating Systems, are the test data sets available for regression testing on the next version?			
f.	Are the detailed test results and reports available for review and audit?			
g.	Do you follow a defined process for tracking the status of all Year 2000 problems reported, changes made, testing, compliance, and return to production?			

9. Commercial Off-the-Shelf (COTS)/Government Off-the-Shelf (GOTS) Components

Please, provide the following information with regard to COTS/GOTS components.

9.		Yes	No	N/A
a.	Does the system use COTS/GOTS application packages and/or infrastructure components?			
b.	If yes, have those items been verified to be Year 2000 compliant?			

9.		Narrative Answer
c.	How was Year 2000 compliance determined? (Certified by vendor or contractor tested in-house, etc.)	

Certification Levels

Certification levels are defined below. Yes, verified and N/A are considered positive responses. No is considered a negative response. Refer to the Compliance Checklist above for line numbers.

Independent Certification

LEVEL	
I	System retired or replaced
II	Full independent testing completed with: - All questions that apply to this system have positive responses
III	Independent audit of system and existing testing completed with: - All questions that apply to this system have positive responses

Self-Certification (CAUTION: Self-certification assumes a higher risk level of potential failures)

IV	Self-certification with full use of 4 digit century date fields - All questions that apply to this system have positive responses
V	Self-certification indicates risk due to use of 2 digit century fields - All questions that apply to this system have positive responses
VI	Self-certification indicates risk due to ambiguous usage of dates - If Question 5-a, b, c or d have negative responses.
VII	Self-certification indicates potential problems (System needs additional work before Year 2000 processing can be assured with any level of reliability) - Question 2-a,b,c or d have negative responses, or - Question 3-a,b,c,d,e,f,g,h,i or j have negative responses, or - Question 4-a,b,c or d have negative responses, or - Question 5-a,b,c or d have negative responses, or - Question 6-a or b have negative responses, or - Question 9-b has a negative response.
VIII	Not certified or not certified yet.

APPENDIX D: ACRONYMS

AIS	-	Automated Information System
ANSI	-	American National Standards Institute
API	-	Application Programming Interface
ASCII	-	American Standard Code for Information Interchange
BIOS	-	Basic input/output system
BPR	-	Business Process Reengineering
CC	-	Century indicator
CIO	-	Chief Information Officer
CIO Council	-	Council of all government agency CIO's.
CFS	-	Center for Standards
CGDN	-	Coast Guard Data Network
COTS	-	Commercial off-the-Shelf
COE	-	Common Operating Environment
CPU	-	Central Processing Unit
DB	-	Database
DBMS	-	Database Management Systems
DOD	-	Department of Defense
DOS	-	Disk Operating System
EC	-	Electronic Commerce
EDI	-	Electronic Data Interchange
ELOC	-	Executable Line of Code
ESU	-	Electronics Support Unit
FIPS	-	Federal Information Processing Standard
GAO	-	General Accounting Office
GOTS	-	Government off-the-Shelf
GPS	-	Global Positioning System
GUI	-	Graphical User Interface
HMI	-	Human Machine Interfaces
IAW	-	In Accordance With
IEEE	-	Institute for Electrical and Electronics Engineers
IPC	-	Interprocess Communication
ISO	-	International Organization for Standardization
LAN	-	Local Area Network
LANIWAN	-	Local Area Network in a Wide Area Network
LED	-	Light Emitting Diode
LOC	-	Lines of Code
MEA	-	Mission Essential Application
MIL-STD	-	Military Standard
MOA	-	Memorandums of Agreement
NIST	-	National Institute of Standards and Technology
O/A	-	Operating Administrator (of a government agency)
O/S	-	Operating System
OMB	-	Office of Management and Budget
OSC	-	Operations Systems Center

OSE	-	Open System Environment
OST	-	Office of the Secretary of Transportation
PM	-	Program Manager
POC	-	Point of Contact
PUB	-	Publication
RAAD	-	Rapid Architecture Application Development
RAD	-	Rapid Application Development
RDBMS	-	Relational Data Base Management System
RTC	-	Remote Terminal Controllers/Real Time Clock
STD	-	Standard
SWII	-	(Coast Guard) Standard Workstation II
SWIII	-	(Coast Guard) Standard Workstation III
TAFIM	-	Technical Architecture Framework for Information Management
TISCOM	-	Telecommunications and Information Systems Command
USCG	-	U. S. Coast Guard
USMTF	-	Uniform Services Message Text Format
WAN	-	Wide Area Network
WWW	-	World Wide Web
Y2K	-	Year 2000

APPENDIX E: GLOSSARY

Calendar errors: errors typically include failing to treat 2000 as a leap year and converting incorrectly between date representations.

Certified system: For purposes of this Plan only, a certified system is a system which the program manager has signed off on as Y2K compliant (Appendix C provides general guidance).

Coast Guard Year 2000 Data Base (CGY2KDB): A tool developed by G-SI/Y2K to support Coast Guard-wide required management information and to provide a migration planning and assessment decision support capability.

Compliant: "compliant" system's dates are stored, manipulated (including, but not limited to calculating, comparing, and sequencing), exchanged, and displayed in a way that cannot be misinterpreted and are not ambiguous. "Compliant" system's hardware and software products' correctly process date and date related data individually and in combination in both the 20th and 21st Centuries. Finally, "compliant" systems have no extended semantics, calendar errors, date overflow, and inconsistent semantics.

Contingency Plan: a plan for responding to the loss of system use due to a disaster such as a flood, fire, computer virus, or major software failure. The plan contains procedures for emergency response, backup, and post-disaster recovery.

Continuity Of Operations Plan (also called business continuity plan) – a plan for continuing to carry out the core functions of an organization despite any level of failure or disruption, either internal or external to the organization.

Conversion: the process of making changes to databases or source code.

Database: an aggregation of data; a file consisting of a number of records or tables, each of which is constructed of files or a particular type, together with a collection of operations that facilitate searching, sorting, recombination, and similar operations.

Data Overflow: many software products represent dates internally as a base date/time plus an offset in days, seconds, or microseconds since that base date/time. Hardware integers holding the offset value can overflow past the maximum corresponding date—an event that may lead to undefined behaviors.

Embedded System: A piece of equipment that contains a microprocessor chip. Many electronic systems and equipment have embedded microchips which contain date-related information (e.g., PBXs, elevators, biomedical equipment, video cameras, photocopiers, vehicles, etc.).

Executable Lines Of Code (ELOC): source lines of code minus comments, white-space and data declarations. The unit of measure used in costing models to capture the effort to create the functional portion of a software program. Used to cost out the effort to develop the functionality.

Extended Semantics: in general, specific values for a date field are reserved for special interpretation. The most common example is interpreting "99" in a 2-digit year field as an indefinite end date, i.e., "does not expire." Another is embedding a date value in a non-date data element.

Independent Validation: A second validation effort to increase confidence that full remediation work has been done on a Coast Guard system by one of three methods: (1) CG System Experts must witness and validate the testing process and test scripts during each validation evolution, (2) CG System Experts recall remediators to step through crucial portions of Y2K Test criteria with them, enough to determine that critical inputs/outputs are consistent and that critical Y2K remediation for that platform has been accomplished, (not a full IV&V), or (3) Employ an Independent Contractor to accomplish that listed within (2) above. Most cost effective and desirable of the three methods of Independent Validation outlined above is (1).

Implementation: according to the OMB definition, the placing of a program back into the production environment after it has been repaired and has successfully completed Y2K validation testing.

Inconsistent Semantics: at interface between systems, software on each side assumes semantics of data passed. Software must make same century assumptions about 2-digit years.

Integration: two or more software applications that must run on the same physical processor(s) and under the same operating system.

Integration Testing: testing to determine that the related information system units perform to specification.

Interface: a boundary across which two systems communicate. An interface might be a hardware connector used to link to other devices, or it might be a convention used to allow communication between two software systems. This is to include interfaces internal to the system, its applications and programs, to other internal or external systems.

Interoperability: (1) The ability of two or more systems or units to exchange data and use information (IEEE STD 610.12) and (2) The ability of two or more systems to exchange information and to mutually use the information that has been exchanged.

Legacy System: An existing automated information system (AIS) or application which will be replaced in whole or part by a migration system.

Line Of Code: a single computer program command, declaration, or instruction. Program size is often measured in lines of code.

Migration System: An existing automated information system (AIS) or application, or a planned and approved AIS or application, that has been officially designated as the single AIS or application to support standard processes for a function.

Mission Critical System: (1) a system that when its capabilities are degraded, the organization realizes a resulting loss of a core capability and (2) has top priority for Y2K fixes.

Non-Mission Critical System: (1) any system other than Mission Critical which is essential to 1 or more commands and (2) has a priority for Y2K fixes lower than any Mission Critical System.

Object Code: the machine code generated by a source code language processor such as an assembler or compiler. A file of object code may be immediately executable or it may require linking with other object code files, e.g. libraries, to produce a complete executable program.

Parallel Processing: where the old and the converted systems are run concurrently.

Parallel Systems. the simultaneous use of more than one computer to solve a problem.

Regression Testing: selective re-testing to detect faults introduced during modification of a system.

Renovation: according to OMB definition, repair or modification of code to enable it to recognize and process dates on/after 1 January 2000 and produce correct data/results.

Risk Assessment: a continuous process performed during all phases of system development to provide an estimate of the damage, loss, or harm that could result from a failure to successfully develop individual system units.

Risk Management: a management approach designed to reduce risks inherent to system development.

System Testing: testing to determine that the results generated by the enterprise's information systems and their units are accurate and the systems perform to specification.

Test Facility: a computer system isolated from the production environment dedicated to the testing and validation of applications and systems units.

Unit Testing: testing to determine that individual program modules perform to specification.

Validation: the process of evaluating a system or unit during or at the end of the development process to determine whether it satisfies specified requirements. In the context of the OMB phases defined in chapter 5, equivalent to system testing.

Year 2000 Compliant: information systems able to accurately process date data--including, but not limited to, calculating, comparing, and sequencing--from, into, and between the twentieth and twenty-first centuries, including leap year calculations.

Year 2000 problem: the potential problems and its variations that might be encountered in any level of computer hardware and software from microcode to application programs, files, and databases that need to correctly interpret year-date data represented in 2-digit-year format.

Year 2000 (Y2K) System: An automated process that uses information technology such as computer hardware and software to perform a specific function, application, or service and is Y2K "Compliant".

APPENDIX F: REFERENCES AND WEB SITES

Office of Architecture and Planning (G-SI/Y2K) brief to the Commandant began the formal Year 2000 awareness phase at Coast Guard Headquarters, dated October 23, 1996.

Director of Information Technology (G-SI) Year 2000 Seminar began the formal awareness phase of Coast Guard five phase management process involving Headquarters Program Managers, Headquarters units, and major field commands, dated December, 12 1996.

(G-SI) Data Call for the Coast Guard Year 2000 Data Base, dated December 20, 1996.

Department of Transportation Year 2000 Assessment Worksheet, dated October 10, 1996.

48 Code of Federal Regulations, Parts 39.002 et. seq.

GAO: Y2K Computing Crisis: An Assessment Guide, Exposure Draft, available at GAO, dated February 1997.

Federal Information Processing Standards (FIPS) 4-1: “Representation for Calendar Date and Ordinal Date for Information Interchange,” dated March 25, 1996.

Office of Management and Budget (OMB) Memorandum on Interagency Information Exchanges of Year Information, dated April 8, 1997.

Key Practices of the Capability Maturity Model, (V1.1), Software Engineering Institute, Carnegie Mellon University, February 1993.

CIO Council Subcommittee on Year 2000 Best Practices, CIO Council Subcommittee on Year 2000, April 1997.

Federal Year 2000 Web Sites

The Uniform Resource Locator (URL) address for various world wide web (www) sites are shown below. Due to changing web addresses, only the upper level (home) page URL is shown for some sites. Users should access this URL and from the home page/index, proceed to the year 2000 (y2k) page, or use a search routine. Other www and CGweb addresses take you directly to the Y2K information page.

The White House Millennium Council: **(Internet)**
    http://www.whitehouse.gov/Initiatives/Millennium/main.shtml

GSA - Year 2000 Information Directory: **(Internet)**
    http://www.itpolicy.gsa.gov/mks/yr2000/y2khome.htm

Coast Guard (Systems Directorate on the **Intranet**):
http://cgweb.hsc.comdt.uscg.mil/g-s/gs.htm

Coast Guard (Systems Directorate on the **Internet**):
http://www.uscg.mil/Systems

Coast Guard Operations Systems Center (OSC):- **(Intranet)**
http://www.osc.uscg.mil

Army: **(Internet)**
http://www.tecom.army.mil/y2k/index.html

Air Force: **(Intranet)**
http://year2000.af.mil/

Navy: **(Internet)**
http://www.doncio.navy.mil/y2k/year2000.htm

Marine Corps: **(Internet)**
http://issb-www1.quantico.usmc.mil/year2000/frames/

Defense Information Systems Agency: **(Internet)**
http://www.disa.mil/cio/y2k/cioosd.html

The Mitre Corporation, USAF Electronic Systems Center (ESC), and Defense Information Systems Agency (DISA): **(Internet)**
http://www.mitre.org/research/y2k/

CIO Council Federal Y2K Commercial Off-the-Shelf (COTS) Product Database: **(Internet)**
http://y2k.policyworks.gov

DOD Test and Evaluation Community Year 2000 Information: **(Internet)**
http://tecnet0.jcte.jcs.mil:9000/htdocs/teinfo/index.htm

GAO Y2K Reports: **(Internet)**
http://www.gao.gov/y2kr.htm

Other Year 2000 Web Sites

The Program Manager's Guide to Software acquisition Best Practice (V1.1), software: Acquisition Best Practices Initiative, Department of Defense 1998: **(Internet)**
http://spmn.com/

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation, International Business Machines Corporation (IBM), April; 1997, **(Internet)**
http://ppdbooks.pok.ibm.com/cgi-bin/bookmgr/bookmgr.cmd/books/y2kpaper/contents

IBM Year 2000 Technical Support Center: **(Internet)**
http://www.software.ibm.com/year2000/

The Year 2000 Information Center: **(Internet)**
http://www.year2000.com

CIO Year 2000 Research Center: **(Internet)**
http://www.cio.com/

State of Wisconsin **(Internet)**
http://y2k.state.wi.us

SHIP2000 **(Internet)**
http://www.ship2000.com/